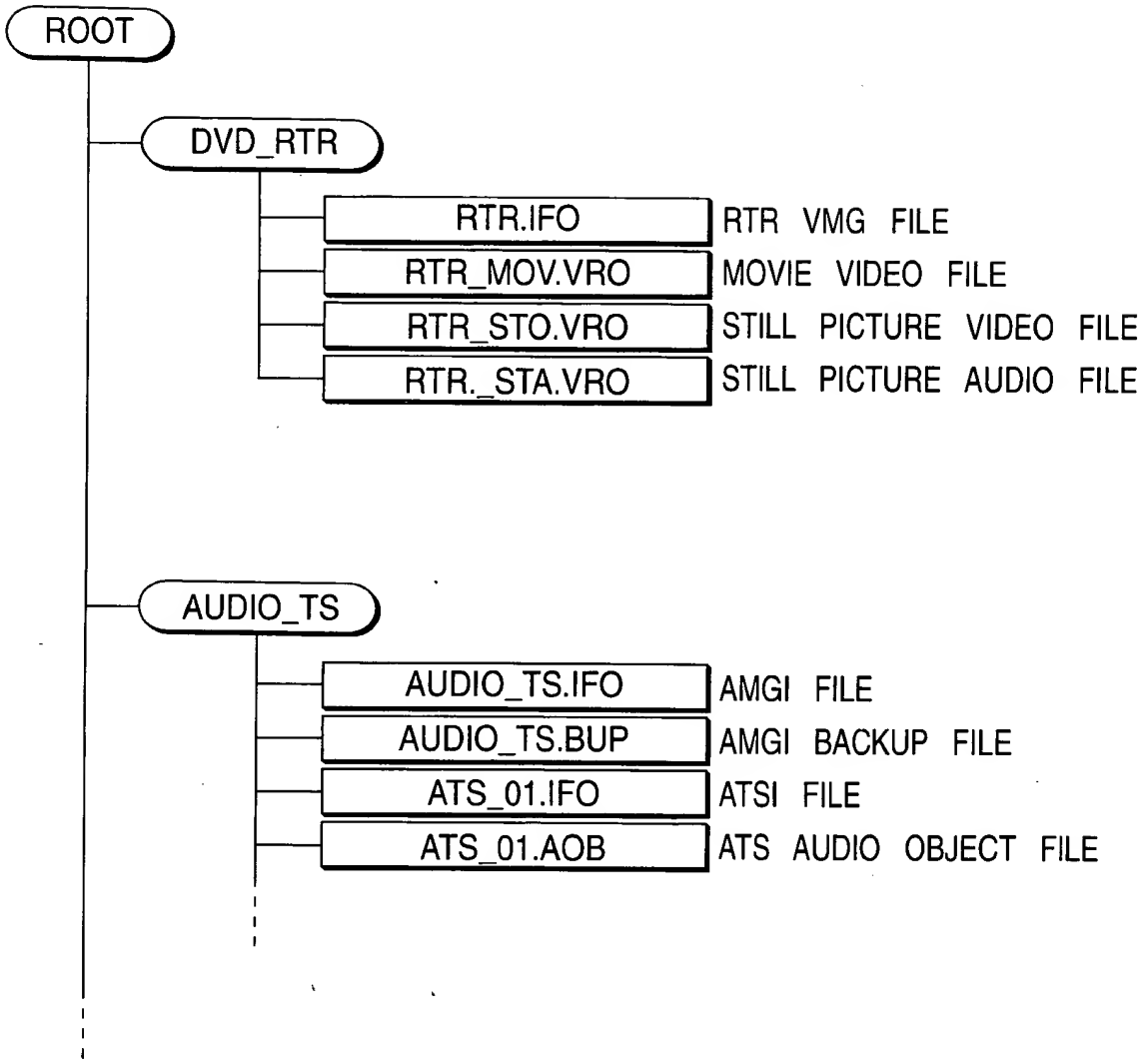

ROOT
DVD_RTR
RTR.IFO
RTR VMG FILE
RTR_MOV.VRO
MOVIE VIDEO FILE
RTR_STO.VRO
STILL PICTURE VIDEO FILE
RTR._STA.VRO
STILL PICTURE AUDIO FILE
AUDIO_TS
AUDIO_TS.IFO
AMGI FILE
AUDIO_TS.BUP
AMGI BACKUP FILE
ATS_01.IFO
ATSI FILE
ATS_01.AOB
ATS AUDIO OBJECT FILE

FIG. 1

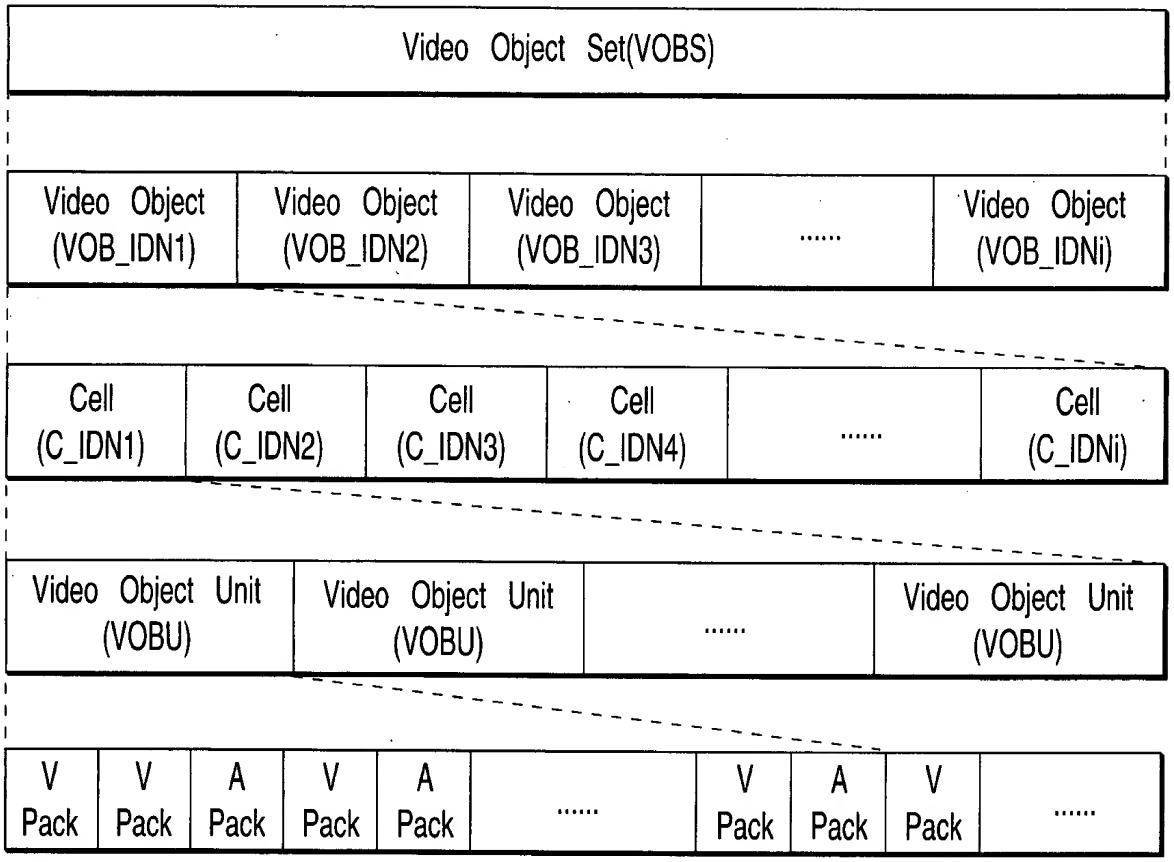


FIG. 2

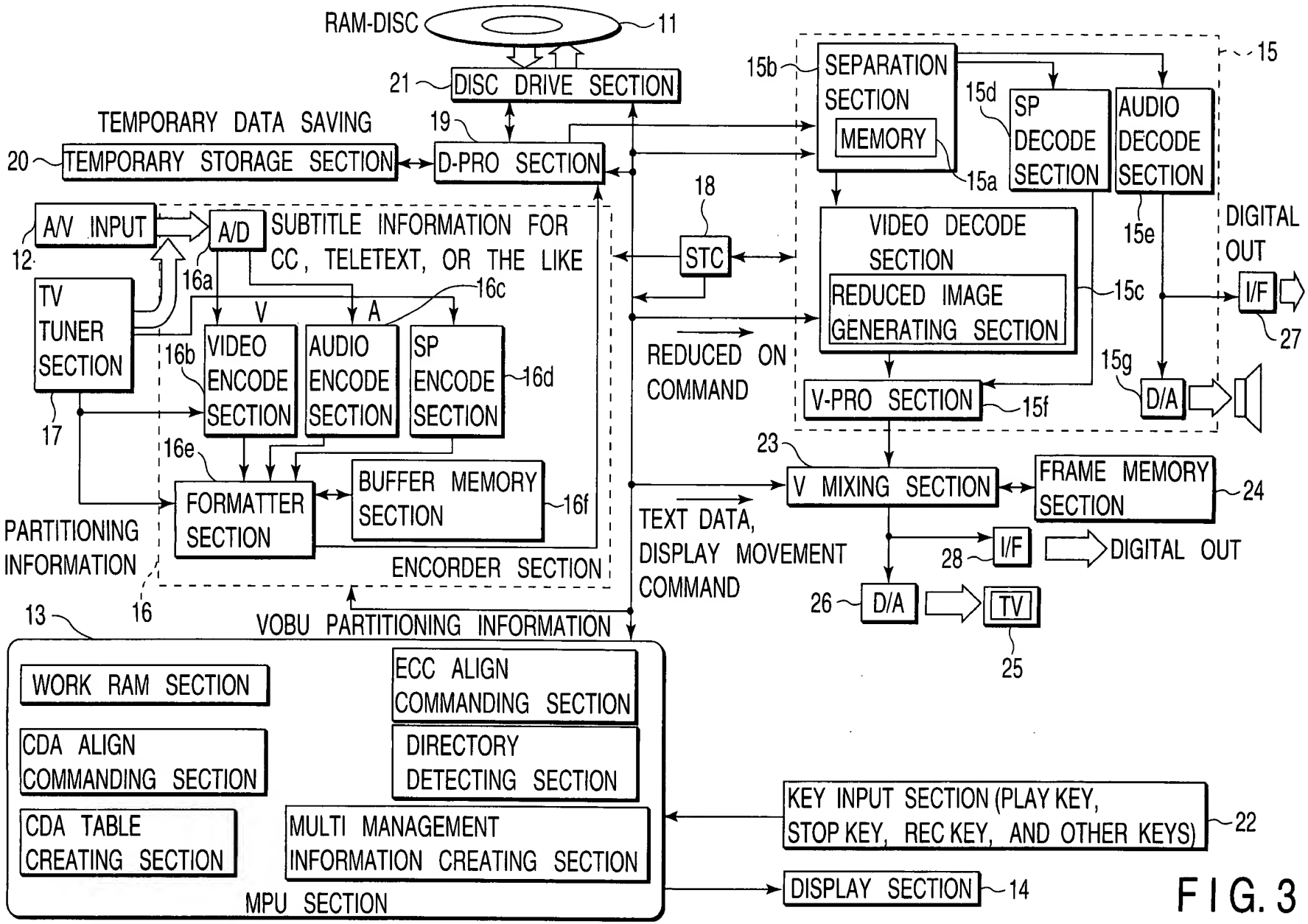


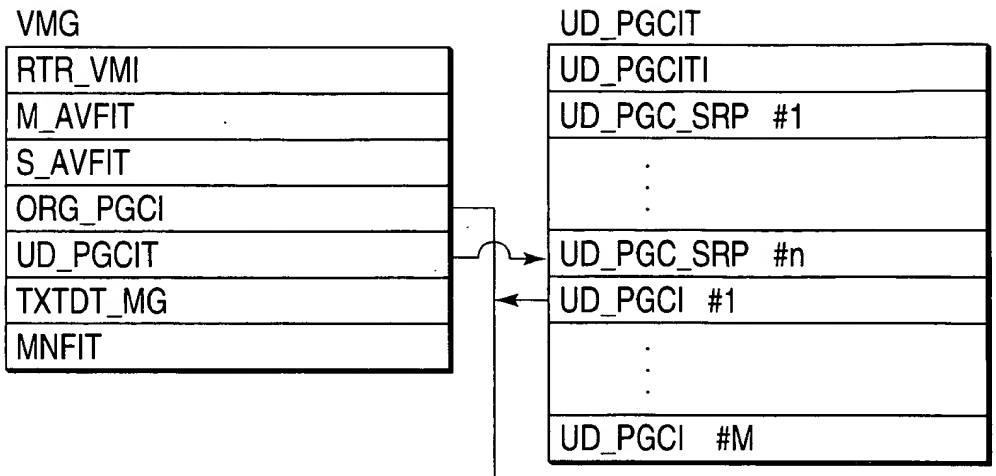
FIG. 3

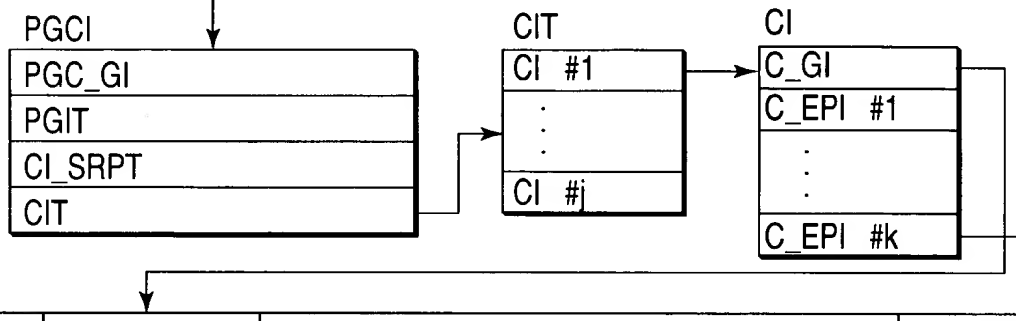
	Byte Number	CDA START ADDRESS: 3 BYTES	CDA SIZE : 2 BYTES	NEXT CDA NUMBER: 2 BYTES
ZONE 0	0	CDA1:0323e(h)	0e00(h)	0002(h)
	7	CDA2:0331e0(h)	0e00(h)	0003(h)
	...	...	...	...
	35	CDA6:0369e0(h)	0e00(h)	0007(h)
ZONE 1	42	CDA7:037d90(h)	0e00(h)	0008(h)
	...	...	...	...
	91	CDA14:03df90(h)	0e00(h)	000F(h)
...	...	...	...	...
ZONE 23	2121	CDA304:158dd0(h)	0e00(h)	0131(h)
	...	...	...	...
	2247	CDA322:1689d0(h)	0e00(h)	0000(h)
	2254	ffffff(h)	ffff(h)	ffff(h)

Byte Number	start CDA Number (2 BYTES)	Byte Number	End Address in End CDA (2 BYTES)
2261	0001(h)	2263	0001(h)

START VOBS WITH CDA WITH START CDA NUMBER AND FOLLOW CDA CHAIN USING NEXT CDA NUMBER. IF NEXT CDA NUMBER IS 00, THIS MEANS END OF VOBS FILE. SUBSEQUENT CDAS REMAIN UNUSED (ONE DISC CONTAINS ONE VOBS FILE). FURTHER, End address in End CDA INDICATES FINAL ADDRESS OF RECORDED DATA SECTOR WITHIN END CDA (RSN RELATIVE TO LEADING CDA)

FIG. 4





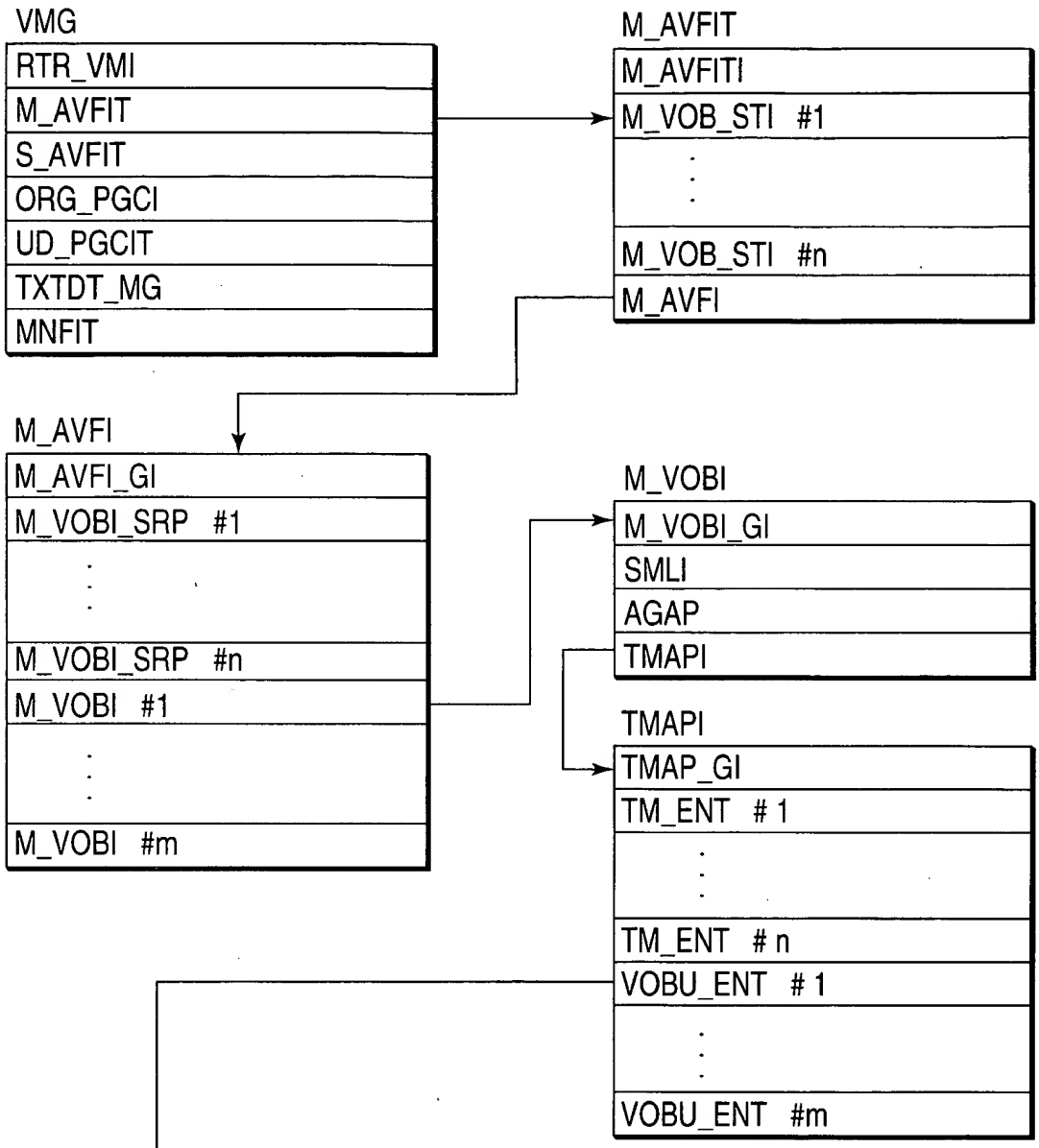
C_GI

RBP		CONTENTS	NUMBER OF BYTES
0 to 0	reserved		
1 to 1	C_TY	CELL Type(00:movie cell 20(h):Still Picture cell	
2 to 3	M_VOBI_SRPN	NUMBER OF SEARCH POINTER OF VOBI	
4 to 5	C_EPI_Ns	NUMBER OF ENTRY POINTS	
6 to 11	C_V_S_PTM	CELL REPRODUCTION START TIME	
12 to 17	C_V_E_PTM	CELL REPRODUCTION END TIME	
		Total	18

C_EPI

RBP		CONTENTS	NUMBER OF BYTES
0 to 0	EP_TY	EP TYPE 0 : PRM_TXTI ABSENT 1: PRM_TXTI PRESENT	
1 to 6	EP_PTM	EP REPRODUCTION TIME	
7 to 134	PRM_TXTI	TEXT INFORMATION	
		Total	135

FIG. 5



VOBU_ENT

RBP		CONTENTS	NUMBER OF BYTES
0 to 0	1STREF_SZ	FINAL ADDRESS OF I PICTURE RELATIVE TO LEADING POSITION OF VOBU	
1 to 1	VOBU_PBTM, VOBU_SZ(HIGH)	TIME REQUIRED TO REPRODUCE VOBU (b7～b2)	
2 to 2	VOBU_SZ(LOW)	VOBU SIZE OF VOBU	
		Total	3 BYTES

FIG. 6

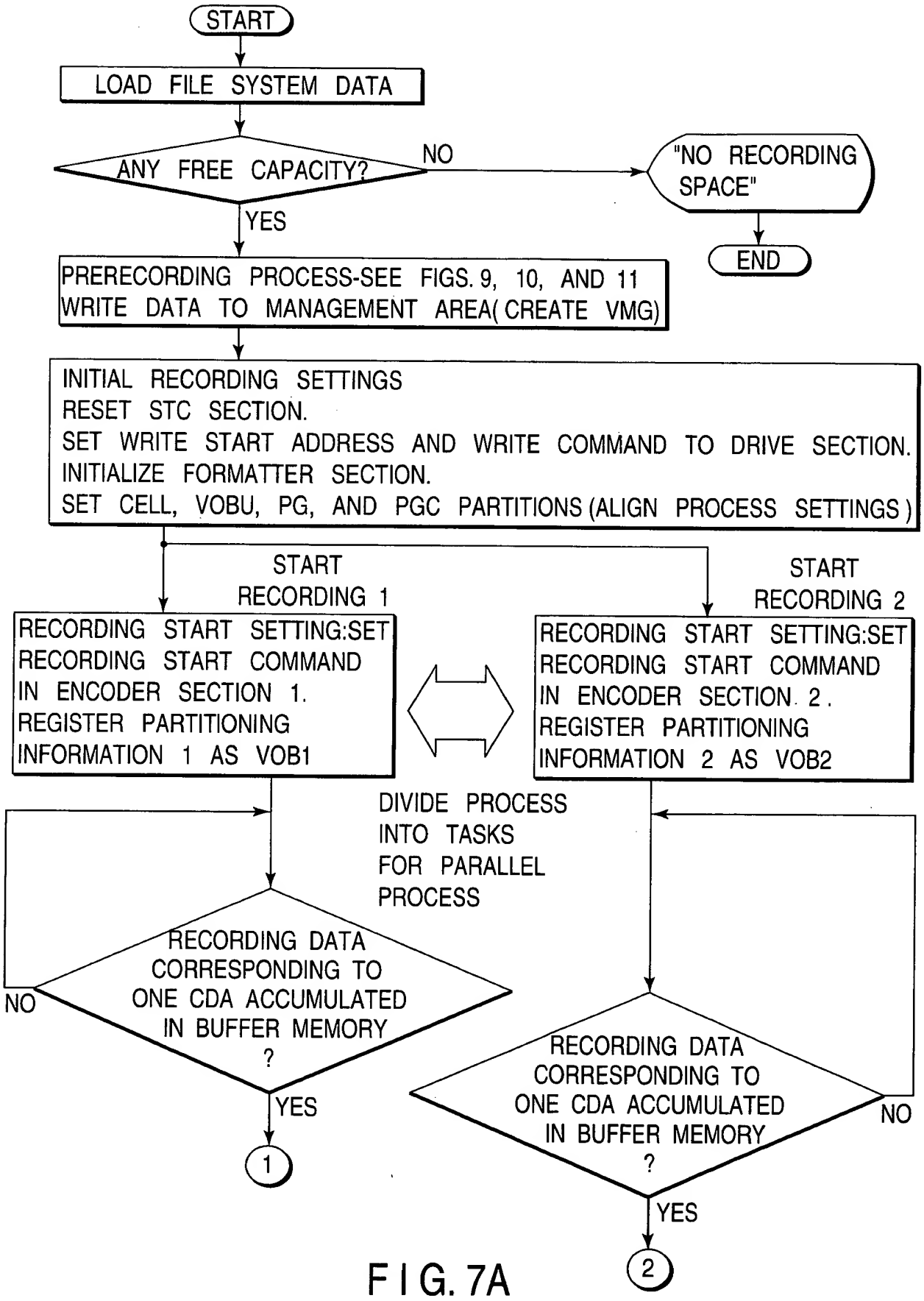

START
LOAD FILE SYSTEM DATA
ANY FREE CAPACITY?
NO
"NO RECORDING SPACE"
END
YES
PRERECORDING PROCESS-SEE FIGS. 9, 10, AND 11
WRITE DATA TO MANAGEMENT AREA(CREATE VMG)
INITIAL RECORDING SETTINGS
RESET STC SECTION.
SET WRITE START ADDRESS AND WRITE COMMAND TO DRIVE SECTION.
INITIALIZE FORMATTER SECTION.
SET CELL, VOBU, PG, AND PGC PARTITIONS (ALIGN PROCESS SETTINGS)
START RECORDING 1
START RECORDING 2
RECORDING START SETTING:SET RECORDING START COMMAND IN ENCODER SECTION 1. REGISTER PARTITIONING INFORMATION 1 AS VOB1
RECORDING START SETTING:SET RECORDING START COMMAND IN ENCODER SECTION 2. REGISTER PARTITIONING INFORMATION 2 AS VOB2
DIVIDE PROCESS INTO TASKS FOR PARALLEL PROCESS
RECORDING DATA CORRESPONDING TO ONE CDA ACCUMULATED IN BUFFER MEMORY ?
NO
YES
1
RECORDING DATA CORRESPONDING TO ONE CDA ACCUMULATED IN BUFFER MEMORY ?
NO
YES
2

FIG. 7A

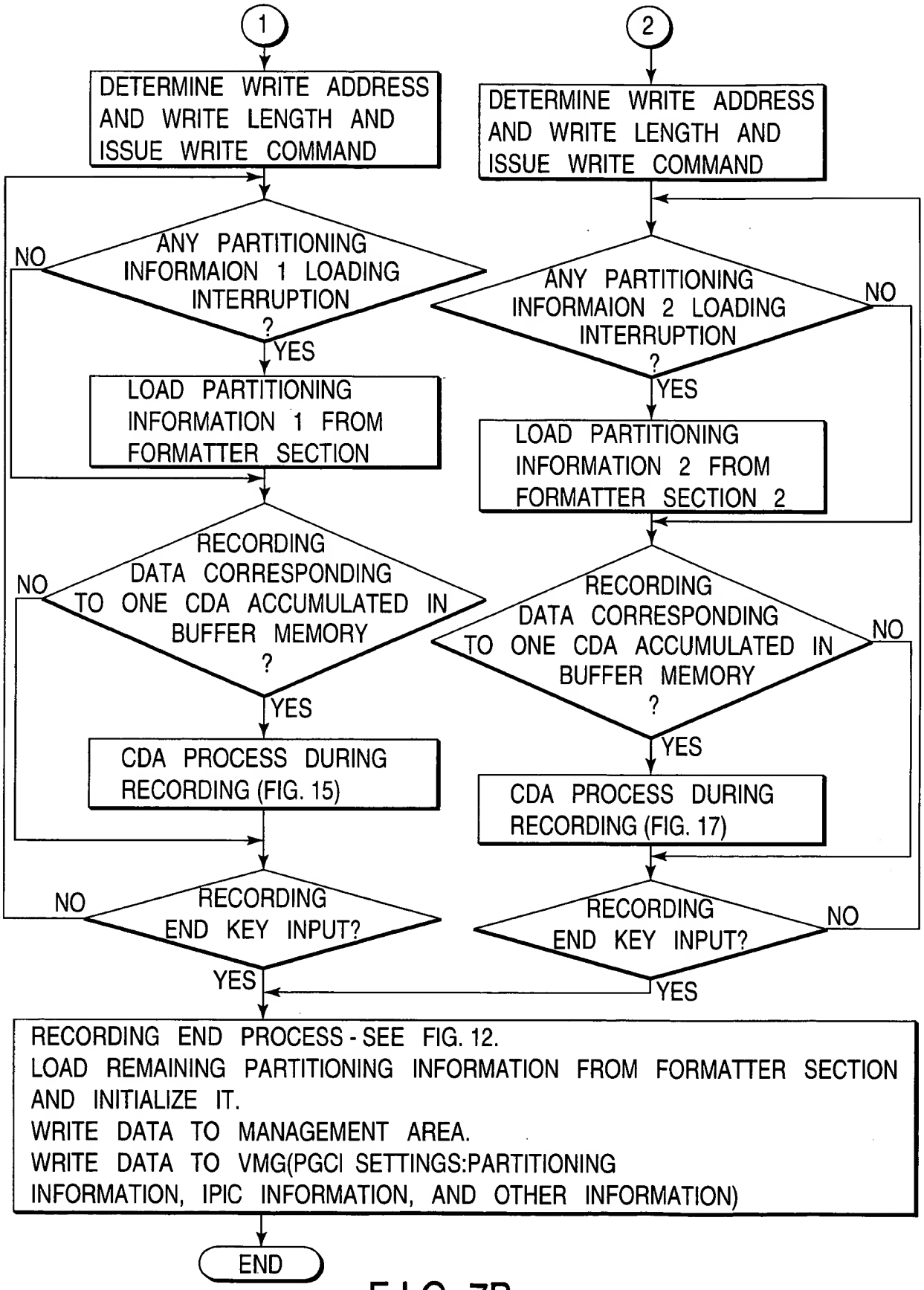

1
2
DETERMINE WRITE ADDRESS AND WRITE LENGTH AND ISSUE WRITE COMMAND
DETERMINE WRITE ADDRESS AND WRITE LENGTH AND ISSUE WRITE COMMAND
ANY PARTITIONING INFORMAION 1 LOADING INTERRUPTION ?
NO
YES
ANY PARTITIONING INFORMAION 2 LOADING INTERRUPTION ?
NO
YES
LOAD PARTITIONING INFORMATION 1 FROM FORMATTER SECTION
LOAD PARTITIONING INFORMATION 2 FROM FORMATTER SECTION 2
RECORDING DATA CORRESPONDING TO ONE CDA ACCUMULATED IN BUFFER MEMORY ?
NO
YES
RECORDING DATA CORRESPONDING TO ONE CDA ACCUMULATED IN BUFFER MEMORY ?
NO
YES
CDA PROCESS DURING RECORDING (FIG. 15)
CDA PROCESS DURING RECORDING (FIG. 17)
RECORDING END KEY INPUT?
NO
YES
RECORDING END KEY INPUT?
NO
YES
RECORDING END PROCESS - SEE FIG. 12.
LOAD REMAINING PARTITIONING INFORMATION FROM FORMATTER SECTION AND INITIALIZE IT.
WRITE DATA TO MANAGEMENT AREA.
WRITE DATA TO VMG(PGCI SETTINGS:PARTITIONING INFORMATION, IPIC INFORMATION, AND OTHER INFORMATION)
END

FIG. 7B

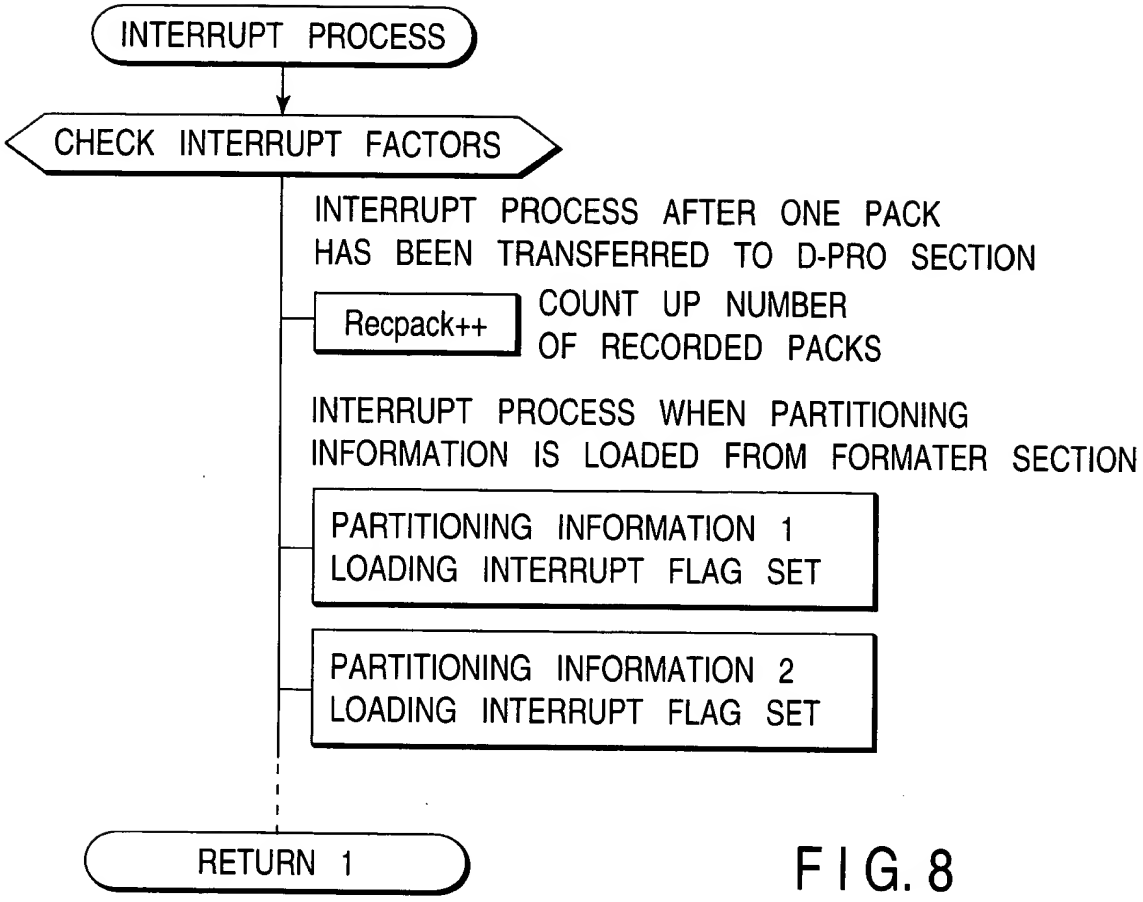

INTERRUPT PROCESS
CHECK INTERRUPT FACTORS
INTERRUPT PROCESS AFTER ONE PACK HAS BEEN TRANSFERRED TO D-PRO SECTION
Recpack++
COUNT UP NUMBER OF RECORDED PACKS
INTERRUPT PROCESS WHEN PARTITIONING INFORMATION IS LOADED FROM FORMATER SECTION
PARTITIONING INFORMATION 1 LOADING INTERRUPT FLAG SET
PARTITIONING INFORMATION 2 LOADING INTERRUPT FLAG SET
RETURN 1

FIG. 8

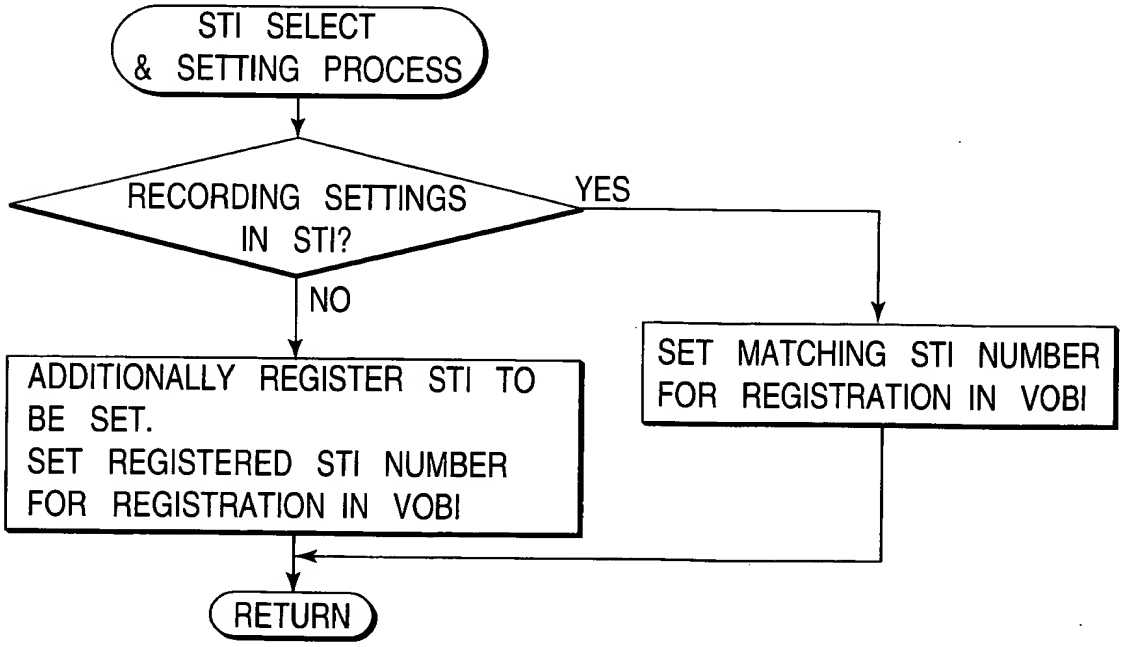

STI SELECT & SETTING PROCESS
RECORDING SETTINGS IN STI?
YES
NO
ADDITIONALLY REGISTER STI TO BE SET.
SET REGISTERED STI NUMBER FOR REGISTRATION IN VOBI
SET MATCHING STI NUMBER FOR REGISTRATION IN VOBI
RETURN

FIG. 10

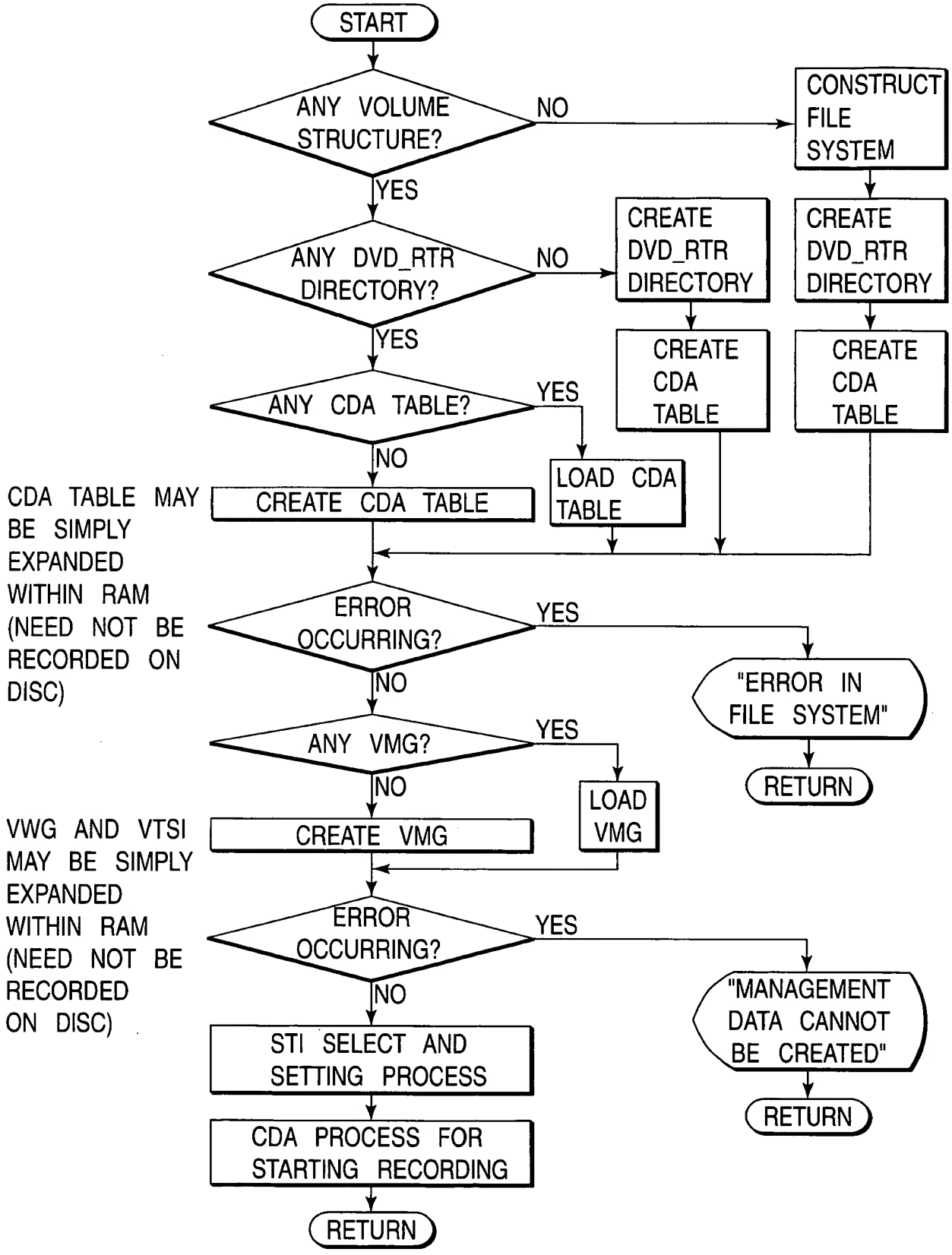

START
ANY VOLUME STRUCTURE?
NO
CONSTRUCT FILE SYSTEM
YES
ANY DVD_RTR DIRECTORY?
NO
CREATE DVD_RTR DIRECTORY
CREATE DVD_RTR DIRECTORY
YES
ANY CDA TABLE?
YES
CREATE CDA TABLE
CREATE CDA TABLE
NO
CREATE CDA TABLE
LOAD CDA TABLE
CDA TABLE MAY BE SIMPLY EXPANDED WITHIN RAM (NEED NOT BE RECORDED ON DISC)
ERROR OCCURRING?
YES
"ERROR IN FILE SYSTEM"
RETURN
NO
ANY VMG?
YES
LOAD VMG
NO
CREATE VMG
VWG AND VTSI MAY BE SIMPLY EXPANDED WITHIN RAM (NEED NOT BE RECORDED ON DISC)
ERROR OCCURRING?
YES
"MANAGEMENT DATA CANNOT BE CREATED"
RETURN
NO
STI SELECT AND SETTING PROCESS
CDA PROCESS FOR STARTING RECORDING
RETURN

FIG. 9

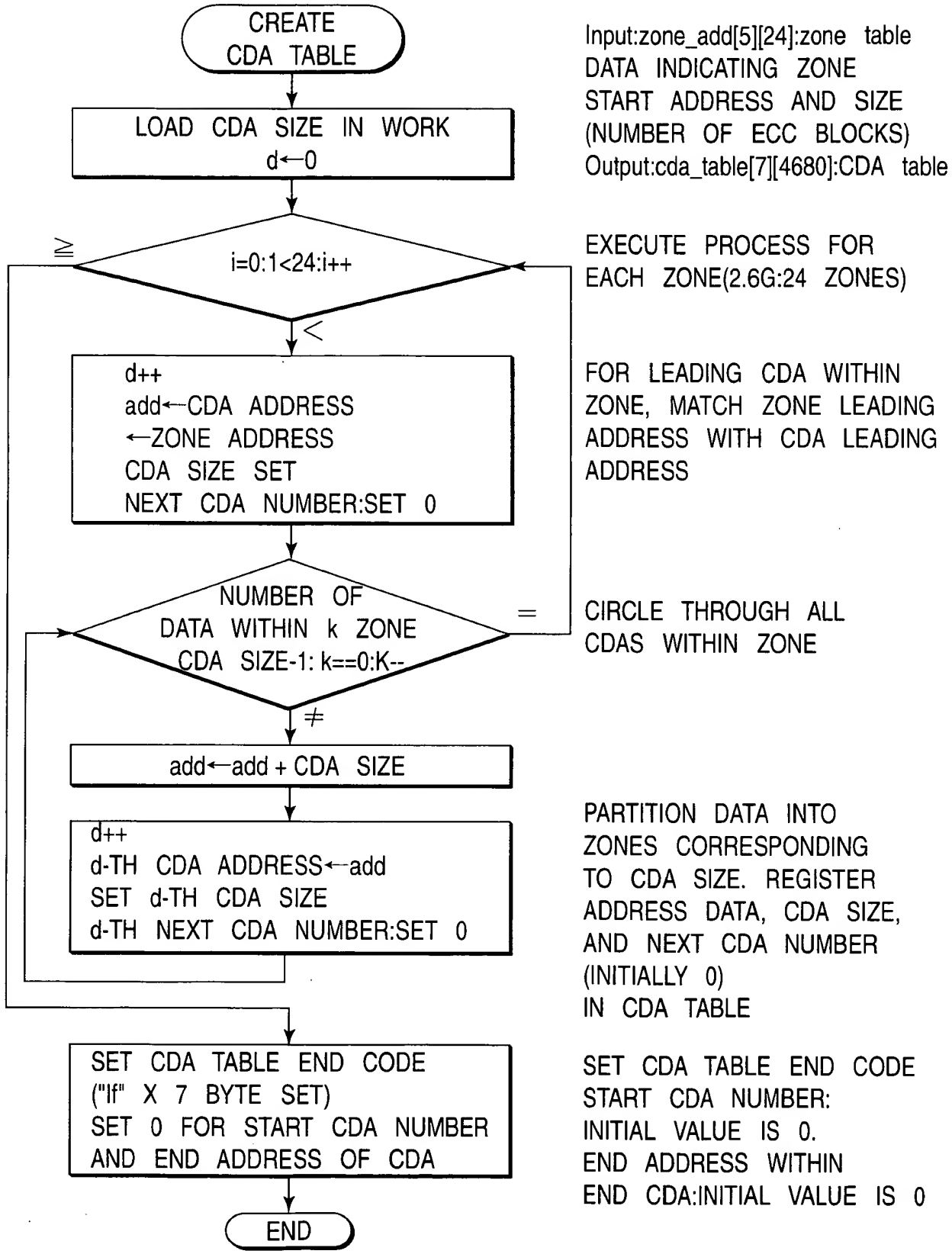

CREATE CDA TABLE
LOAD CDA SIZE IN WORK
d←0
i=0:1<24:i++
≧
<
d++
add←CDA ADDRESS
←ZONE ADDRESS
CDA SIZE SET
NEXT CDA NUMBER:SET 0
NUMBER OF DATA WITHIN k ZONE CDA SIZE-1: k==0:K--
=
≠
add←add + CDA SIZE
d++
d-TH CDA ADDRESS←add
SET d-TH CDA SIZE
d-TH NEXT CDA NUMBER:SET 0
SET CDA TABLE END CODE
("If" X 7 BYTE SET)
SET 0 FOR START CDA NUMBER
AND END ADDRESS OF CDA
END
Input:zone_add[5][24]:zone table
DATA INDICATING ZONE
START ADDRESS AND SIZE
(NUMBER OF ECC BLOCKS)
Output:cda_table[7][4680]:CDA table
EXECUTE PROCESS FOR
EACH ZONE(2.6G:24 ZONES)
FOR LEADING CDA WITHIN
ZONE, MATCH ZONE LEADING
ADDRESS WITH CDA LEADING
ADDRESS
CIRCLE THROUGH ALL
CDAS WITHIN ZONE
PARTITION DATA INTO
ZONES CORRESPONDING
TO CDA SIZE. REGISTER
ADDRESS DATA, CDA SIZE,
AND NEXT CDA NUMBER
(INITIALLY 0)
IN CDA TABLE
SET CDA TABLE END CODE
START CDA NUMBER:
INITIAL VALUE IS 0.
END ADDRESS WITHIN
END CDA:INITIAL VALUE IS 0

FIG. 11

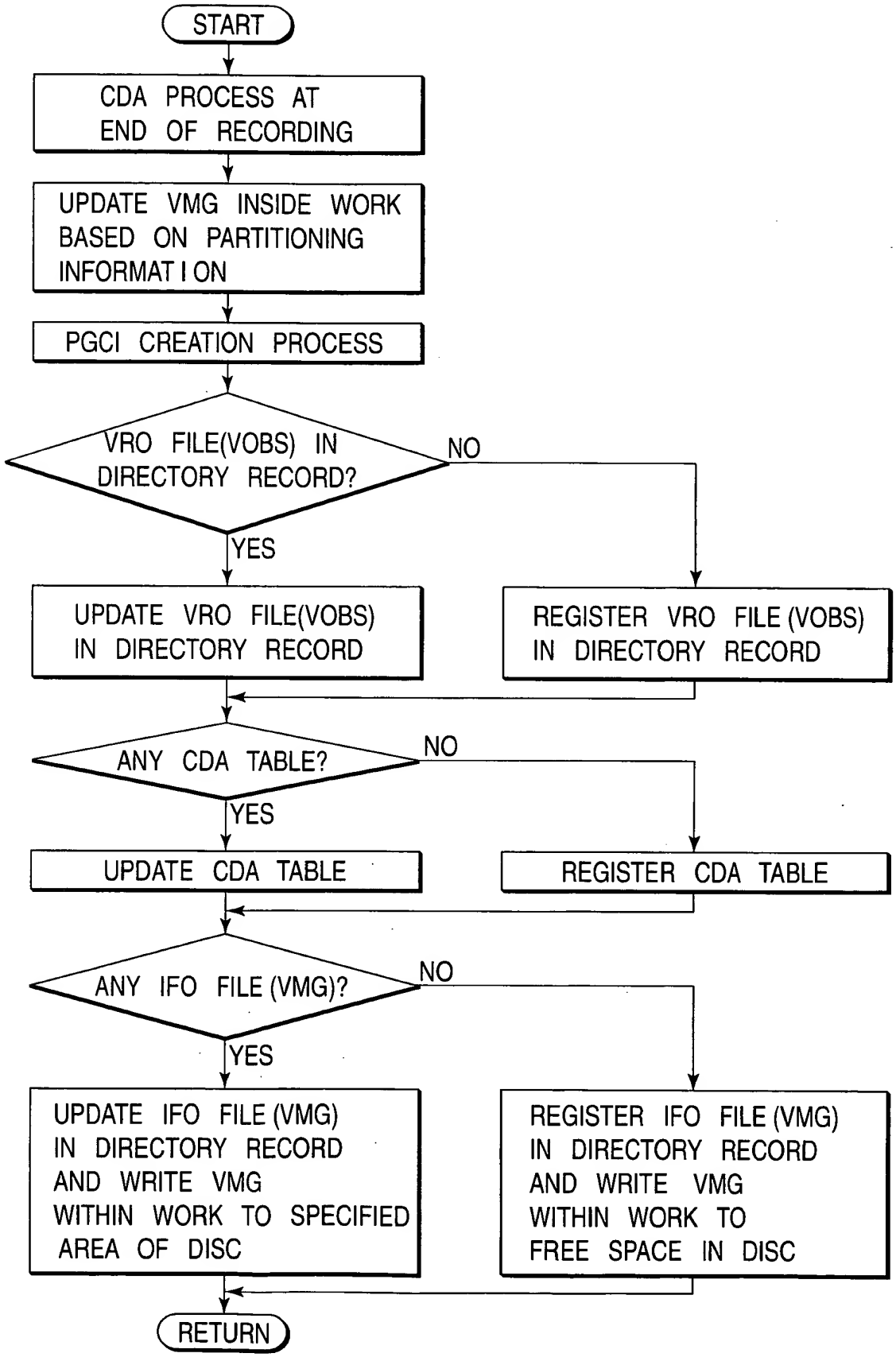

START
CDA PROCESS AT END OF RECORDING
UPDATE VMG INSIDE WORK BASED ON PARTITIONING INFORMATION
PGCI CREATION PROCESS
VRO FILE(VOBS) IN DIRECTORY RECORD?
NO
YES
UPDATE VRO FILE(VOBS) IN DIRECTORY RECORD
REGISTER VRO FILE (VOBS) IN DIRECTORY RECORD
ANY CDA TABLE?
NO
YES
UPDATE CDA TABLE
REGISTER CDA TABLE
ANY IFO FILE (VMG)?
NO
YES
UPDATE IFO FILE (VMG) IN DIRECTORY RECORD AND WRITE VMG WITHIN WORK TO SPECIFIED AREA OF DISC
REGISTER IFO FILE (VMG) IN DIRECTORY RECORD AND WRITE VMG WITHIN WORK TO FREE SPACE IN DISC
RETURN

FIG. 12

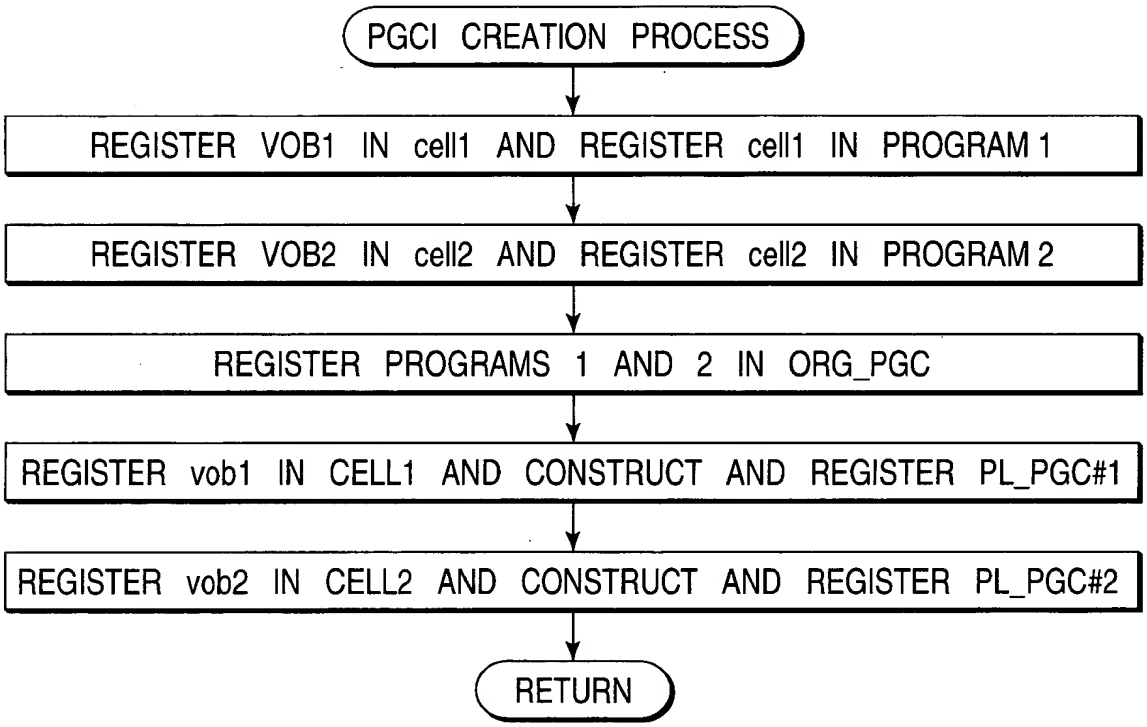

PGCI CREATION PROCESS
REGISTER VOB1 IN cell1 AND REGISTER cell1 IN PROGRAM 1
REGISTER VOB2 IN cell2 AND REGISTER cell2 IN PROGRAM 2
REGISTER PROGRAMS 1 AND 2 IN ORG_PGC
REGISTER vob1 IN CELL1 AND CONSTRUCT AND REGISTER PL_PGC#1
REGISTER vob2 IN CELL2 AND CONSTRUCT AND REGISTER PL_PGC#2
RETURN

FIG. 13

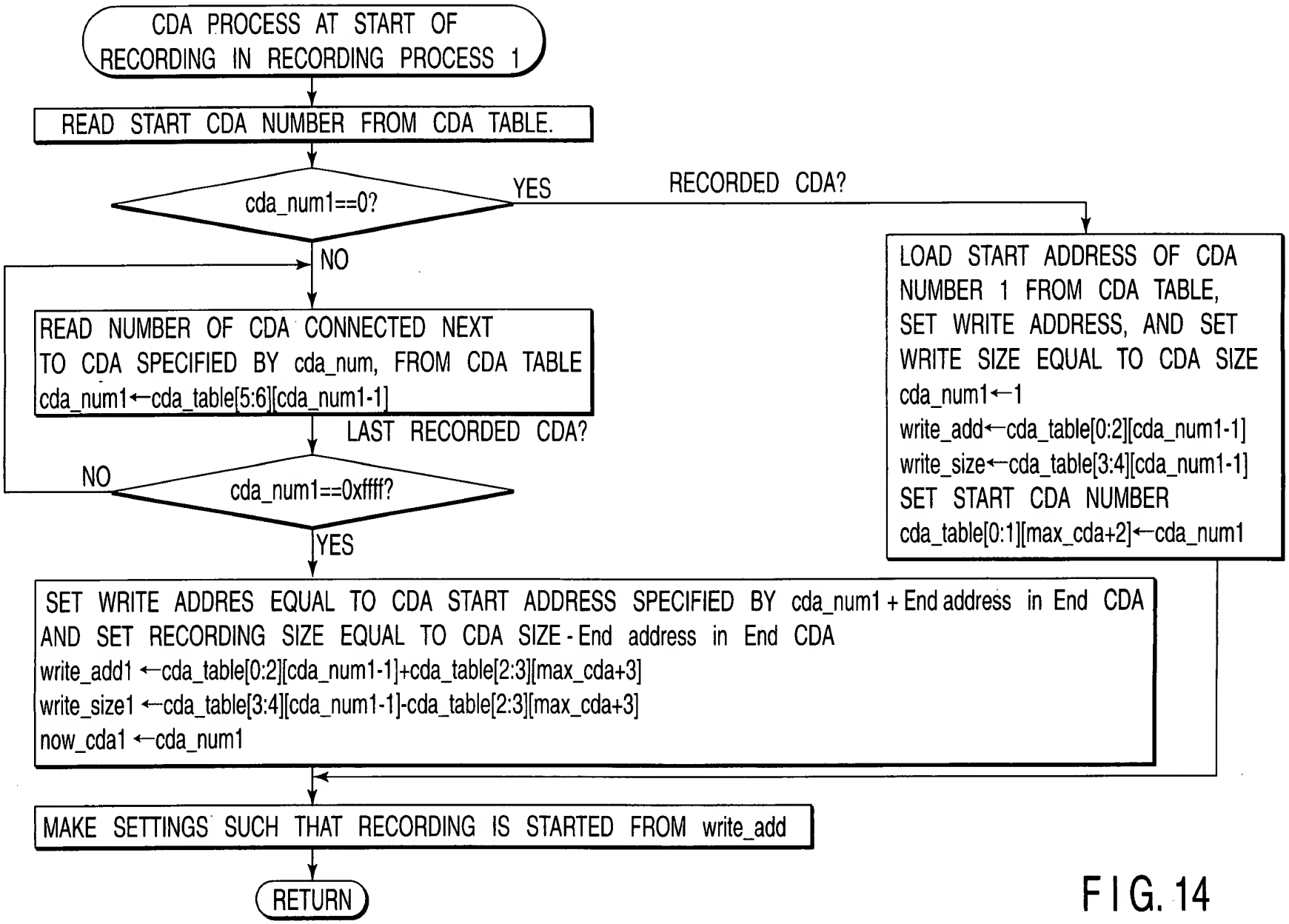

CDA PROCESS AT START OF RECORDING IN RECORDING PROCESS 1
READ START CDA NUMBER FROM CDA TABLE.
cda_num1==0?
YES
RECORDED CDA?
NO
READ NUMBER OF CDA CONNECTED NEXT TO CDA SPECIFIED BY cda_num, FROM CDA TABLE
cda_num1←cda_table[5:6][cda_num1-1]
LAST RECORDED CDA?
NO
cda_num1==0xffff?
YES
LOAD START ADDRESS OF CDA NUMBER 1 FROM CDA TABLE, SET WRITE ADDRESS, AND SET WRITE SIZE EQUAL TO CDA SIZE
cda_num1←1
write_add←cda_table[0:2][cda_num1-1]
write_size←cda_table[3:4][cda_num1-1]
SET START CDA NUMBER
cda_table[0:1][max_cda+2]←cda_num1
SET WRITE ADDRES EQUAL TO CDA START ADDRESS SPECIFIED BY cda_num1 + End address in End CDA AND SET RECORDING SIZE EQUAL TO CDA SIZE - End address in End CDA
write_add1 ←cda_table[0:2][cda_num1-1]+cda_table[2:3][max_cda+3]
write_size1 ←cda_table[3:4][cda_num1-1]-cda_table[2:3][max_cda+3]
now_cda1 ←cda_num1
MAKE SETTINGS SUCH THAT RECORDING IS STARTED FROM write_add
RETURN

FIG. 14

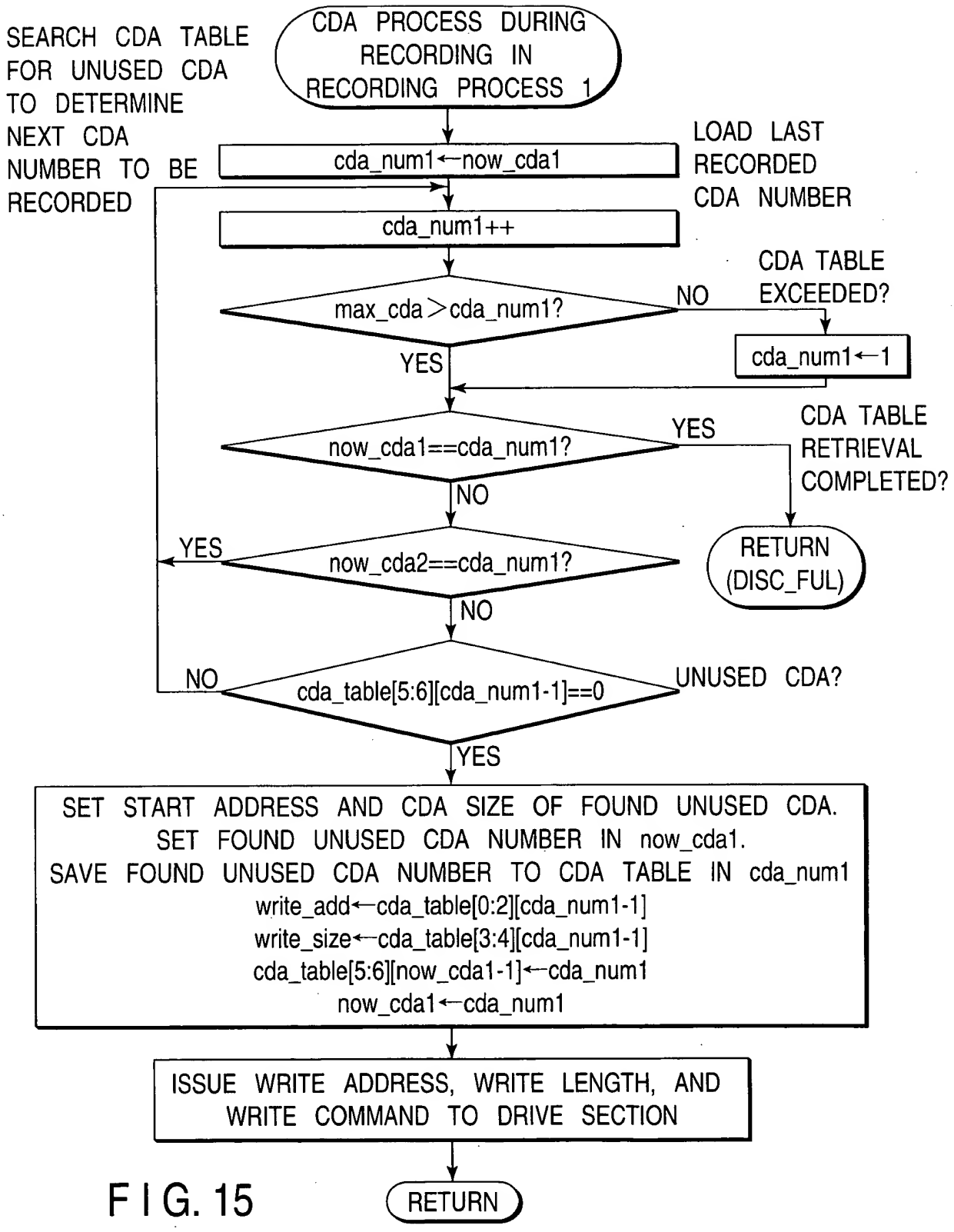

CDA PROCESS DURING RECORDING IN RECORDING PROCESS 1
SEARCH CDA TABLE FOR UNUSED CDA TO DETERMINE NEXT CDA NUMBER TO BE RECORDED
cda_num1←now_cda1
LOAD LAST RECORDED CDA NUMBER
cda_num1++
max_cda>cda_num1?
NO
CDA TABLE EXCEEDED?
cda_num1←1
YES
now_cda1==cda_num1?
YES
CDA TABLE RETRIEVAL COMPLETED?
NO
now_cda2==cda_num1?
YES
RETURN (DISC_FUL)
NO
cda_table[5:6][cda_num1-1]==0
NO
UNUSED CDA?
YES
SET START ADDRESS AND CDA SIZE OF FOUND UNUSED CDA.
SET FOUND UNUSED CDA NUMBER IN now_cda1.
SAVE FOUND UNUSED CDA NUMBER TO CDA TABLE IN cda_num1
write_add←cda_table[0:2][cda_num1-1]
write_size←cda_table[3:4][cda_num1-1]
cda_table[5:6][now_cda1-1]←cda_num1
now_cda1←cda_num1
ISSUE WRITE ADDRESS, WRITE LENGTH, AND WRITE COMMAND TO DRIVE SECTION
RETURN

FIG. 15

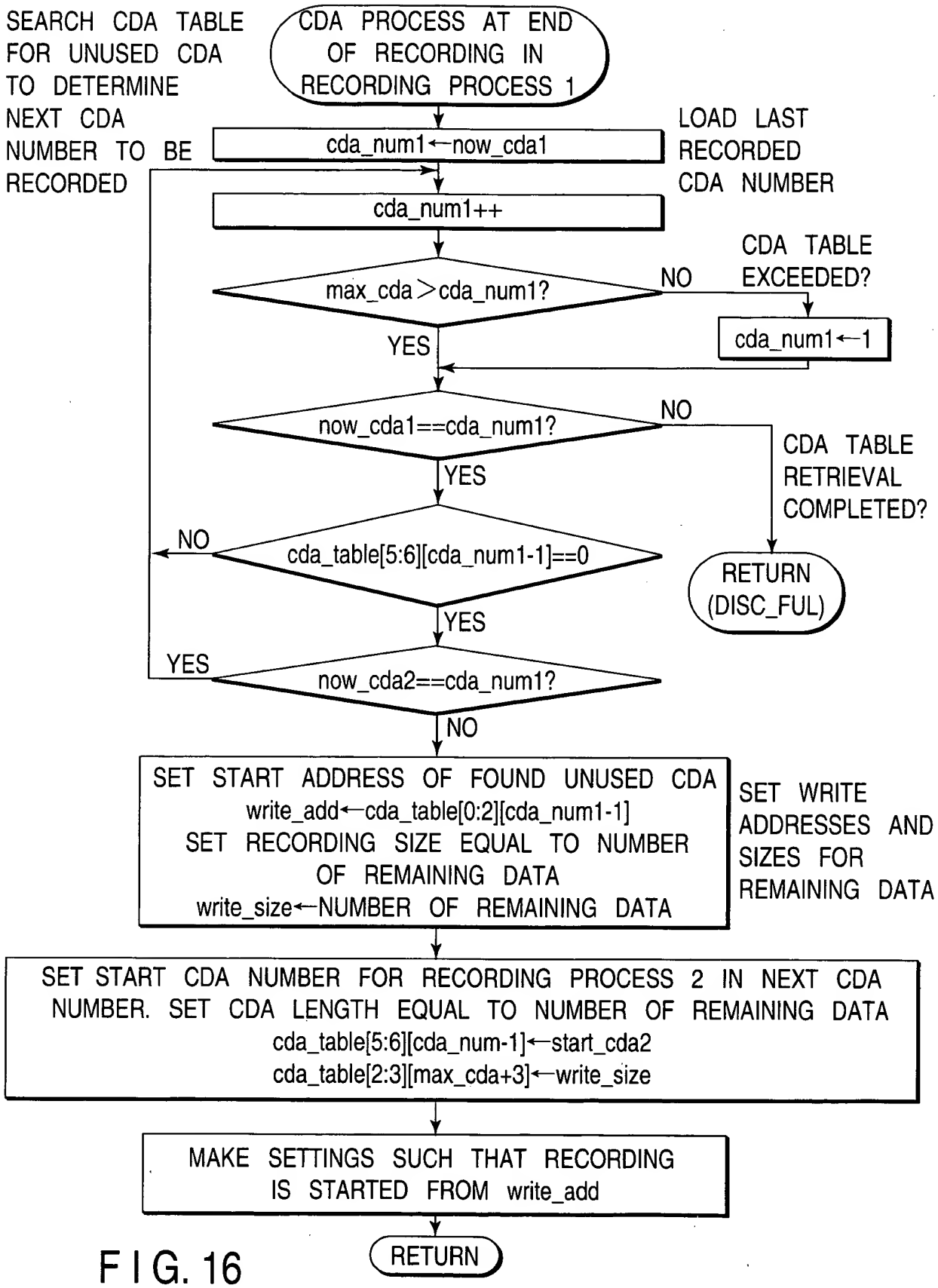

SEARCH CDA TABLE FOR UNUSED CDA TO DETERMINE NEXT CDA NUMBER TO BE RECORDED
CDA PROCESS AT END OF RECORDING IN RECORDING PROCESS 1
cda_num1←now_cda1
LOAD LAST RECORDED CDA NUMBER
cda_num1++
max_cda>cda_num1?
NO
CDA TABLE EXCEEDED?
cda_num1←1
YES
now_cda1==cda_num1?
NO
CDA TABLE RETRIEVAL COMPLETED?
YES
cda_table[5:6][cda_num1-1]==0
NO
RETURN (DISC_FUL)
YES
now_cda2==cda_num1?
YES
NO
SET START ADDRESS OF FOUND UNUSED CDA
write_add←cda_table[0:2][cda_num1-1]
SET RECORDING SIZE EQUAL TO NUMBER OF REMAINING DATA
write_size←NUMBER OF REMAINING DATA
SET WRITE ADDRESSES AND SIZES FOR REMAINING DATA
SET START CDA NUMBER FOR RECORDING PROCESS 2 IN NEXT CDA NUMBER. SET CDA LENGTH EQUAL TO NUMBER OF REMAINING DATA
cda_table[5:6][cda_num-1]←start_cda2
cda_table[2:3][max_cda+3]←write_size
MAKE SETTINGS SUCH THAT RECORDING IS STARTED FROM write_add
RETURN

FIG. 16

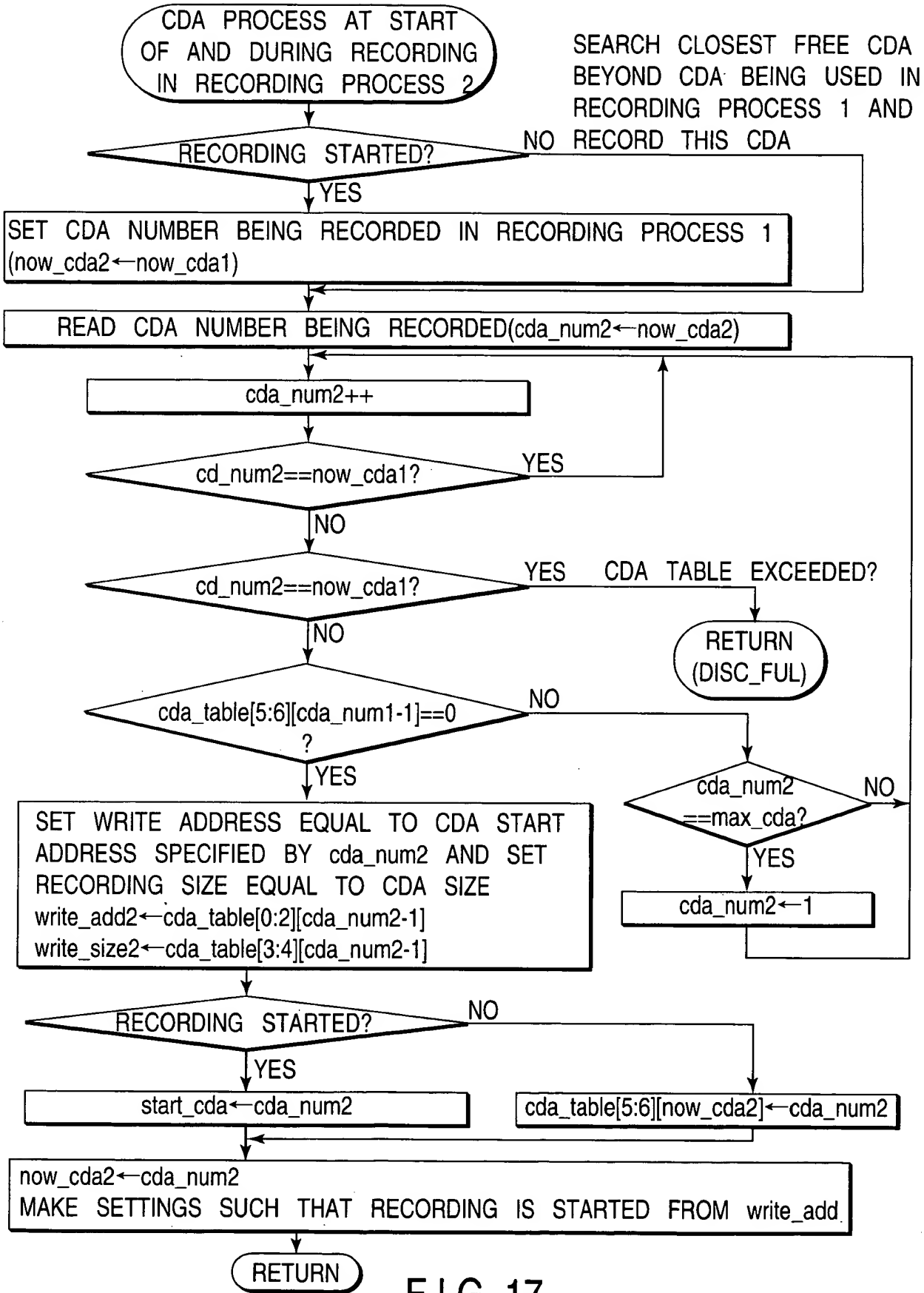

CDA PROCESS AT START OF AND DURING RECORDING IN RECORDING PROCESS 2
SEARCH CLOSEST FREE CDA BEYOND CDA BEING USED IN RECORDING PROCESS 1 AND RECORD THIS CDA
RECORDING STARTED?
NO
YES
SET CDA NUMBER BEING RECORDED IN RECORDING PROCESS 1 (now_cda2←now_cda1)
READ CDA NUMBER BEING RECORDED(cda_num2←now_cda2)
cda_num2++
cd_num2==now_cda1?
YES
NO
cd_num2==now_cda1?
YES
CDA TABLE EXCEEDED?
NO
RETURN (DISC_FUL)
cda_table[5:6][cda_num1-1]==0 ?
NO
YES
cda_num2 ==max_cda?
NO
YES
cda_num2←1
SET WRITE ADDRESS EQUAL TO CDA START ADDRESS SPECIFIED BY cda_num2 AND SET RECORDING SIZE EQUAL TO CDA SIZE
write_add2←cda_table[0:2][cda_num2-1]
write_size2←cda_table[3:4][cda_num2-1]
RECORDING STARTED?
NO
YES
start_cda←cda_num2
cda_table[5:6][now_cda2]←cda_num2
now_cda2←cda_num2
MAKE SETTINGS SUCH THAT RECORDING IS STARTED FROM write_add
RETURN

FIG. 17

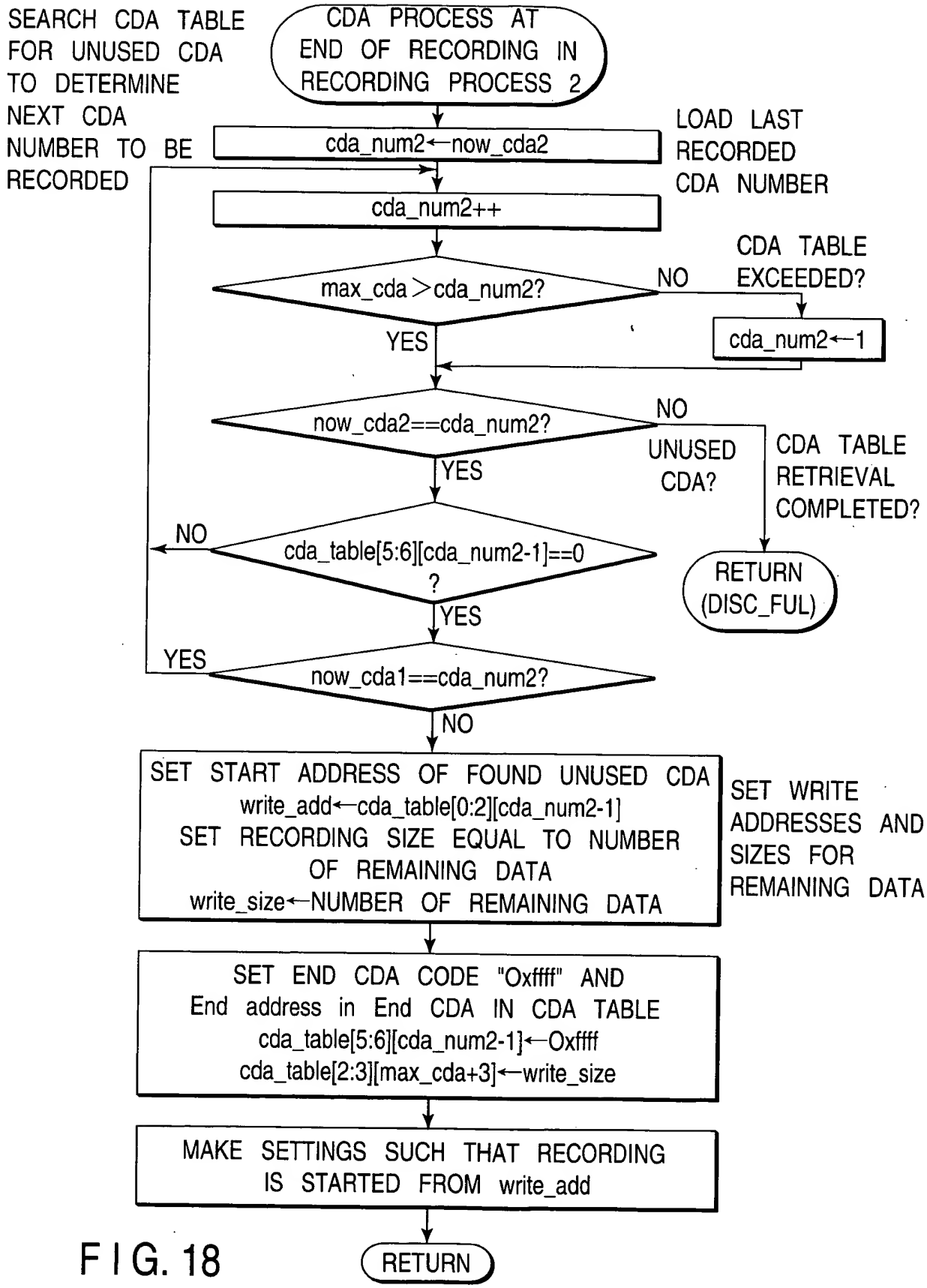
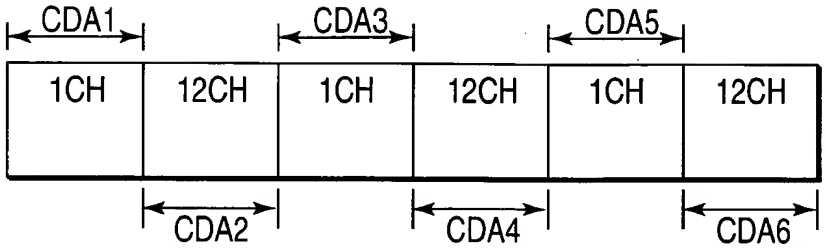
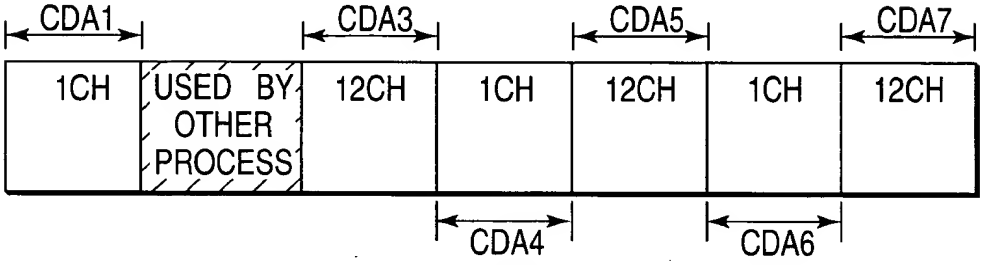

SEARCH CDA TABLE FOR UNUSED CDA TO DETERMINE NEXT CDA NUMBER TO BE RECORDED
CDA PROCESS AT END OF RECORDING IN RECORDING PROCESS 2
cda_num2←now_cda2
LOAD LAST RECORDED CDA NUMBER
cda_num2++
max_cda>cda_num2?
NO
CDA TABLE EXCEEDED?
cda_num2←1
YES
now_cda2==cda_num2?
NO
UNUSED CDA?
CDA TABLE RETRIEVAL COMPLETED?
YES
NO
cda_table[5:6][cda_num2-1]==0 ?
RETURN (DISC_FUL)
YES
YES
now_cda1==cda_num2?
NO
SET START ADDRESS OF FOUND UNUSED CDA
write_add←cda_table[0:2][cda_num2-1]
SET RECORDING SIZE EQUAL TO NUMBER OF REMAINING DATA
write_size←NUMBER OF REMAINING DATA
SET WRITE ADDRESSES AND SIZES FOR REMAINING DATA
SET END CDA CODE "Oxffff" AND End address in End CDA IN CDA TABLE
cda_table[5:6][cda_num2-1]←Oxffff
cda_table[2:3][max_cda+3]←write_size
MAKE SETTINGS SUCH THAT RECORDING IS STARTED FROM write_add
RETURN

FIG. 18



EXAMPLE OF RECORDING IN UNUSED DISC

FIG. 19A



EXAPMLE OF DATA BEING PARTLY USED

FIG. 19B

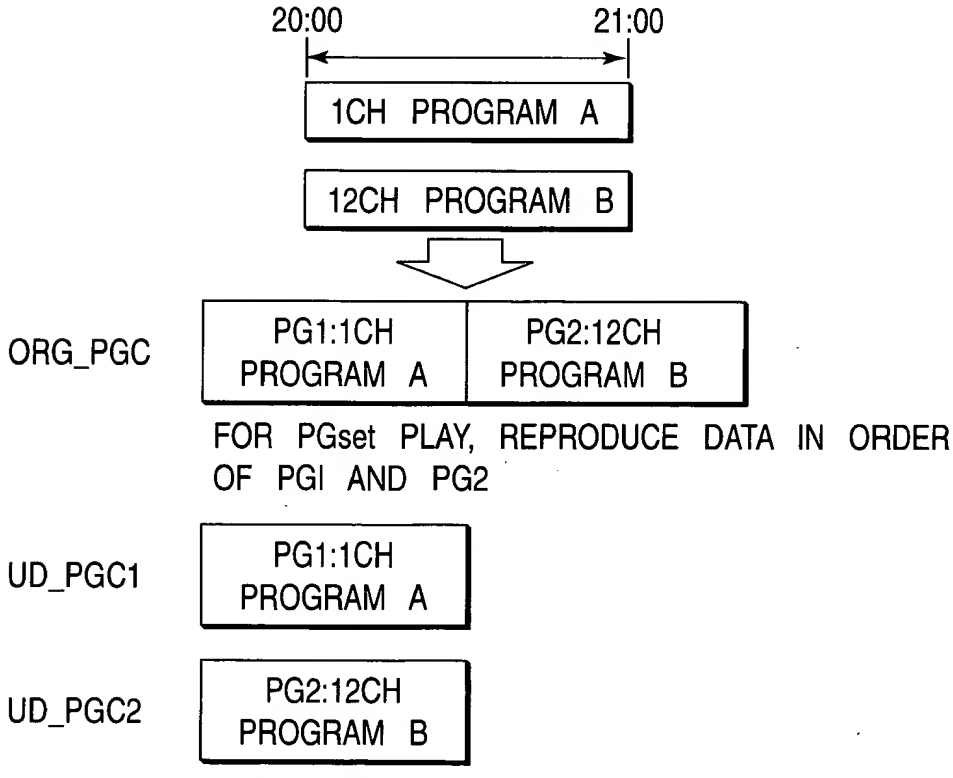


FIG. 20

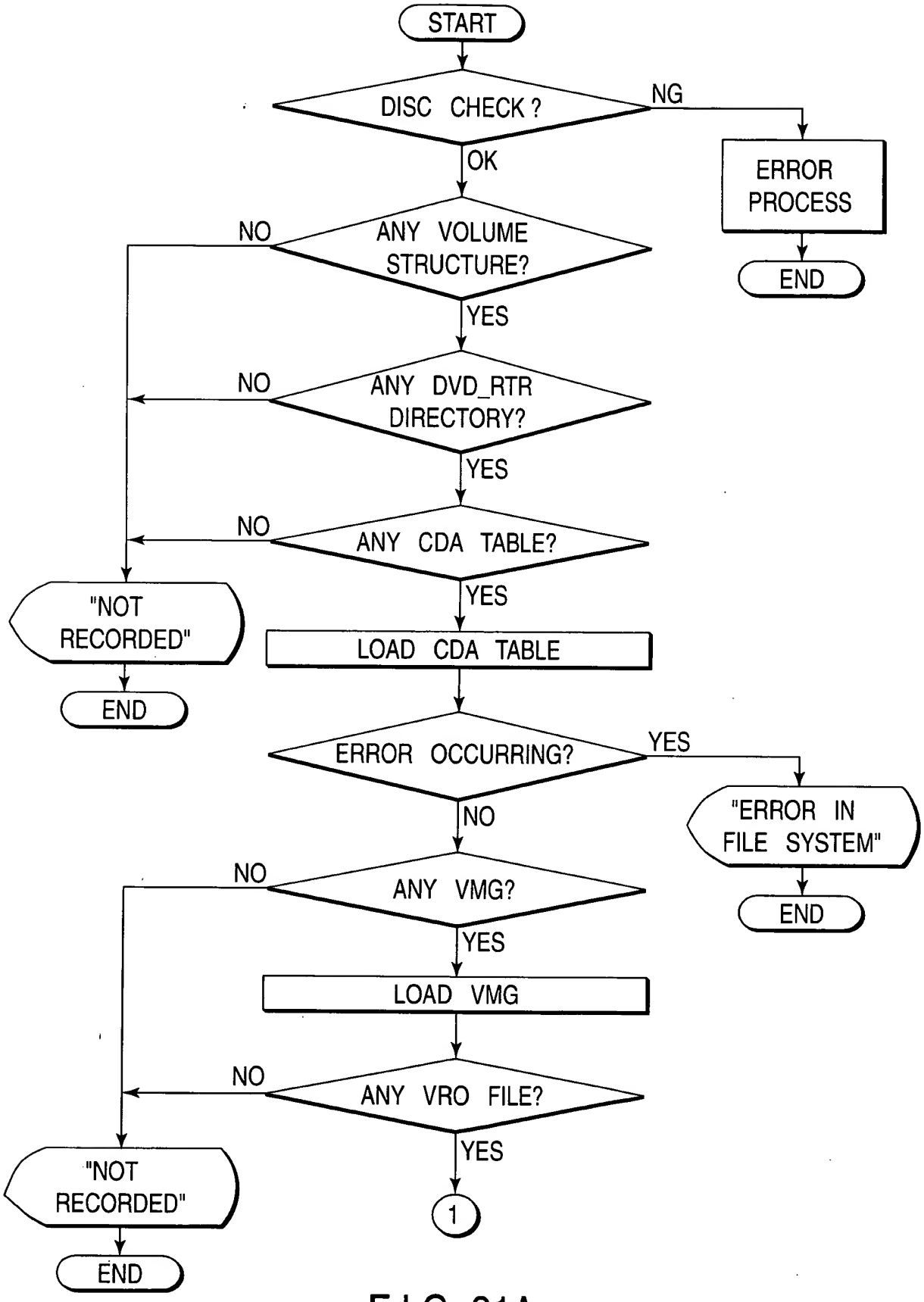

START
DISC CHECK ?
NG
ERROR PROCESS
END
OK
NO
ANY VOLUME STRUCTURE?
YES
NO
ANY DVD_RTR DIRECTORY?
YES
NO
ANY CDA TABLE?
YES
"NOT RECORDED"
END
LOAD CDA TABLE
ERROR OCCURRING?
YES
"ERROR IN FILE SYSTEM"
END
NO
NO
ANY VMG?
YES
LOAD VMG
NO
ANY VRO FILE?
YES
"NOT RECORDED"
END
1

FIG. 21A

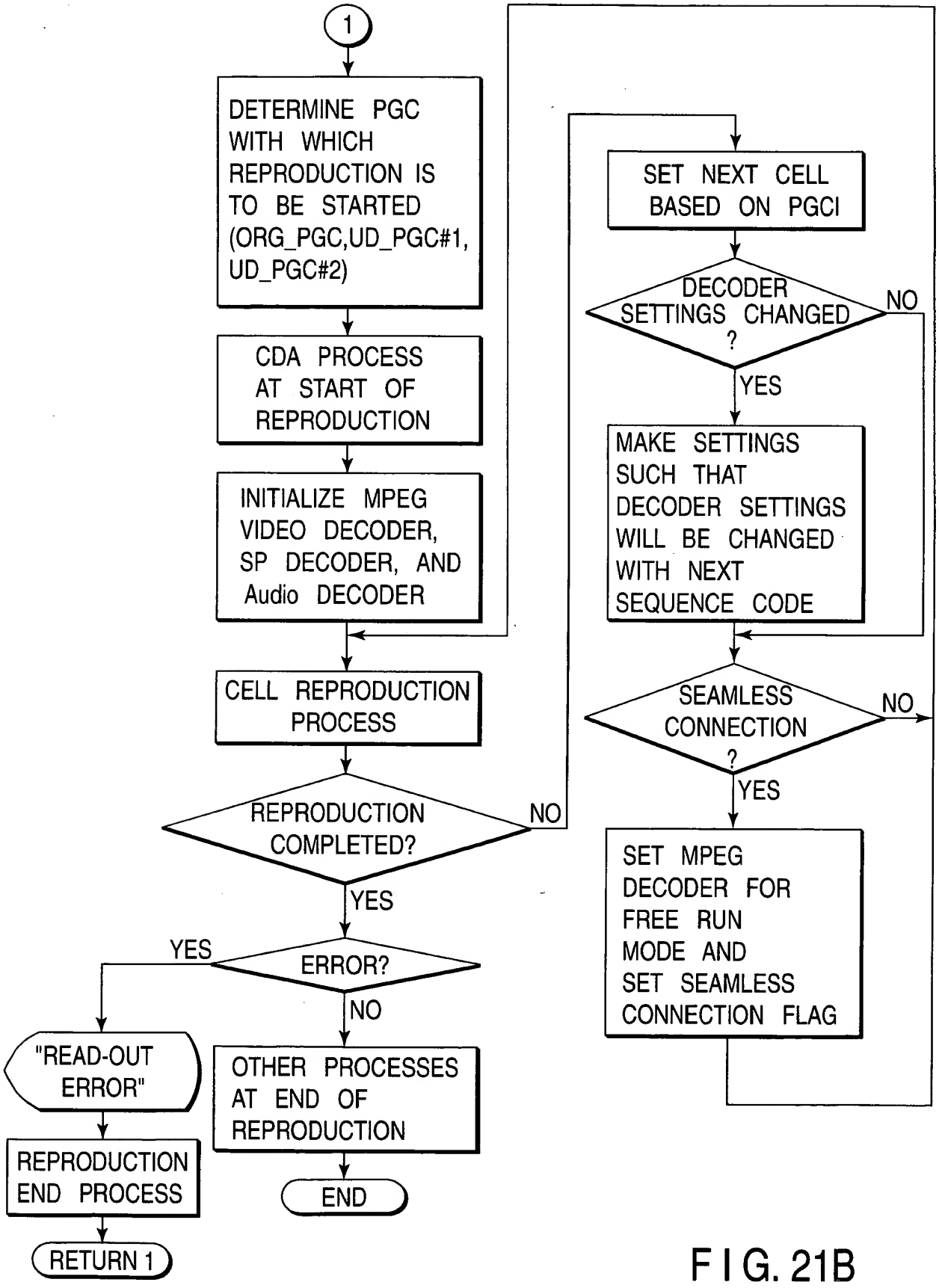

1
DETERMINE PGC WITH WHICH REPRODUCTION IS TO BE STARTED (ORG_PGC,UD_PGC#1, UD_PGC#2)
CDA PROCESS AT START OF REPRODUCTION
INITIALIZE MPEG VIDEO DECODER, SP DECODER, AND Audio DECODER
CELL REPRODUCTION PROCESS
REPRODUCTION COMPLETED?
NO
YES
ERROR?
YES
NO
"READ-OUT ERROR"
REPRODUCTION END PROCESS
RETURN 1
OTHER PROCESSES AT END OF REPRODUCTION
END
SET NEXT CELL BASED ON PGCI
DECODER SETTINGS CHANGED ?
NO
YES
MAKE SETTINGS SUCH THAT DECODER SETTINGS WILL BE CHANGED WITH NEXT SEQUENCE CODE
SEAMLESS CONNECTION ?
NO
YES
SET MPEG DECODER FOR FREE RUN MODE AND SET SEAMLESS CONNECTION FLAG

FIG. 21B

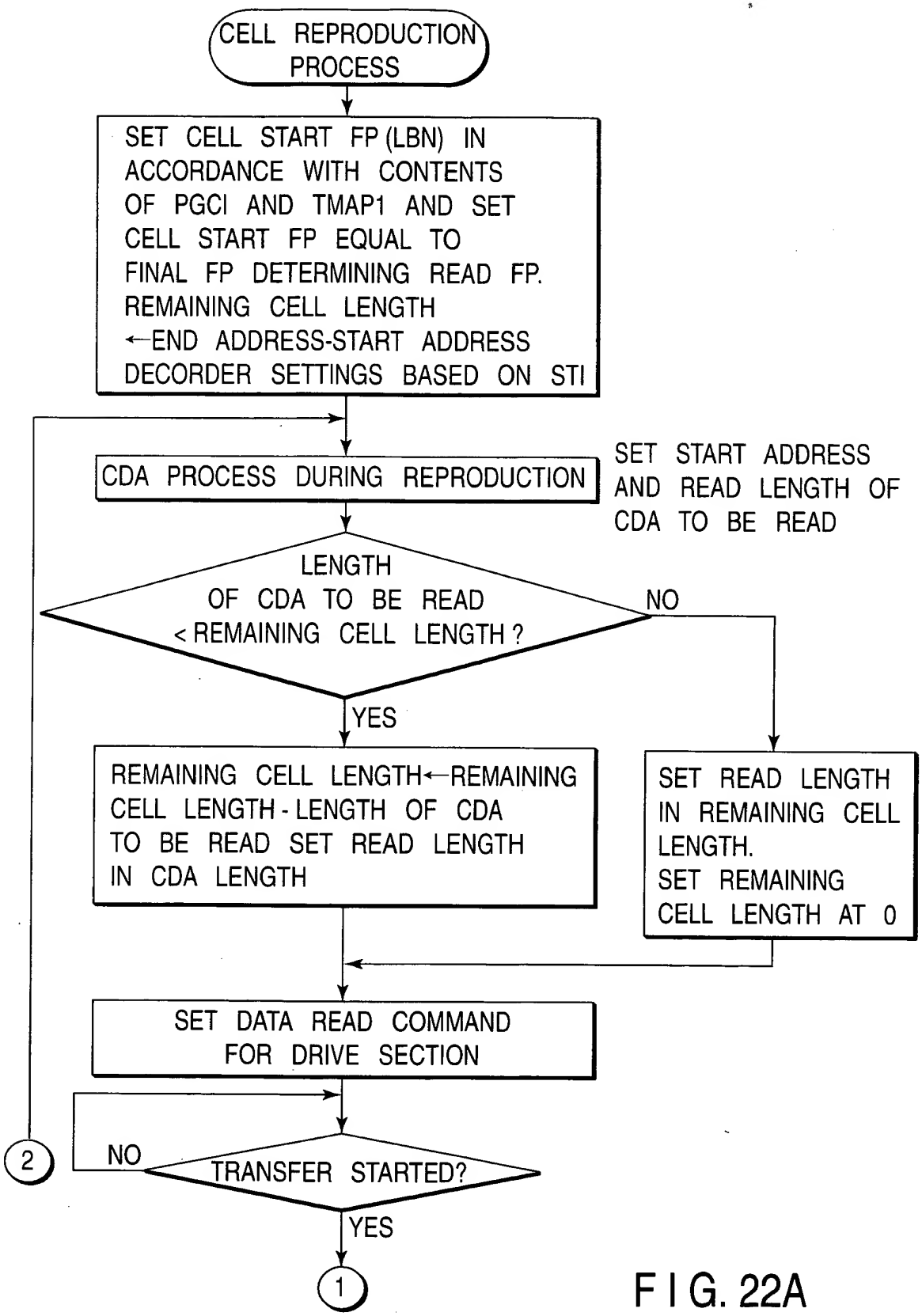

CELL REPRODUCTION PROCESS
SET CELL START FP (LBN) IN ACCORDANCE WITH CONTENTS OF PGCI AND TMAP1 AND SET CELL START FP EQUAL TO FINAL FP DETERMINING READ FP. REMAINING CELL LENGTH ←END ADDRESS-START ADDRESS DECORDER SETTINGS BASED ON STI
CDA PROCESS DURING REPRODUCTION
SET START ADDRESS AND READ LENGTH OF CDA TO BE READ
LENGTH OF CDA TO BE READ < REMAINING CELL LENGTH ?
NO
YES
REMAINING CELL LENGTH←REMAINING CELL LENGTH - LENGTH OF CDA TO BE READ SET READ LENGTH IN CDA LENGTH
SET READ LENGTH IN REMAINING CELL LENGTH. SET REMAINING CELL LENGTH AT 0
SET DATA READ COMMAND FOR DRIVE SECTION
NO
TRANSFER STARTED?
YES
2
1

FIG. 22A

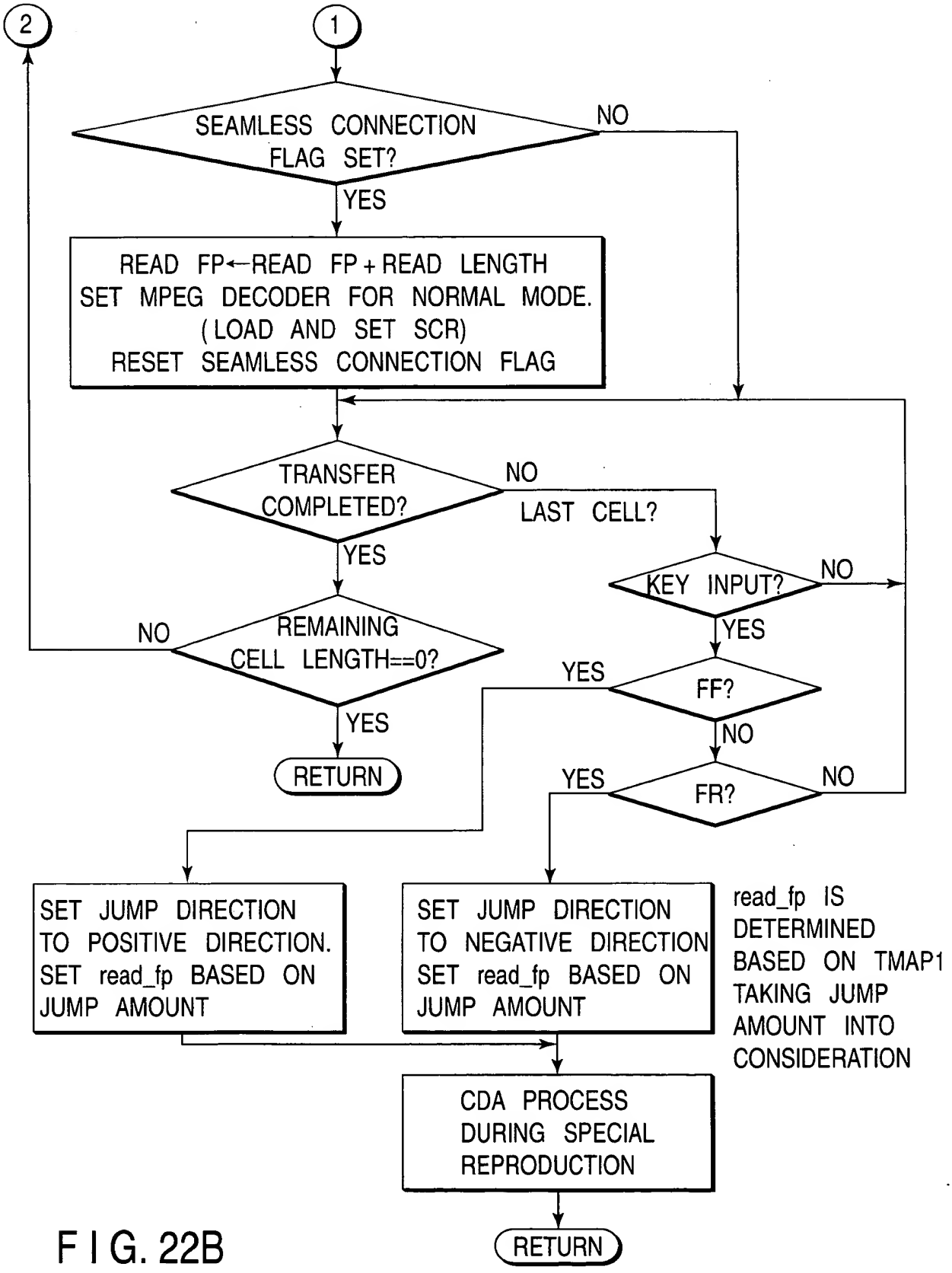

2
1
SEAMLESS CONNECTION FLAG SET?
NO
YES
READ FP←READ FP + READ LENGTH
SET MPEG DECODER FOR NORMAL MODE.
(LOAD AND SET SCR)
RESET SEAMLESS CONNECTION FLAG
TRANSFER COMPLETED?
NO
LAST CELL?
YES
KEY INPUT?
NO
YES
REMAINING CELL LENGTH==0?
NO
YES
FF?
YES
NO
RETURN
FR?
YES
NO
SET JUMP DIRECTION TO POSITIVE DIRECTION. SET read_fp BASED ON JUMP AMOUNT
SET JUMP DIRECTION TO NEGATIVE DIRECTION SET read_fp BASED ON JUMP AMOUNT
read_fp IS DETERMINED BASED ON TMAP1 TAKING JUMP AMOUNT INTO CONSIDERATION
CDA PROCESS DURING SPECIAL REPRODUCTION
RETURN

FIG. 22B

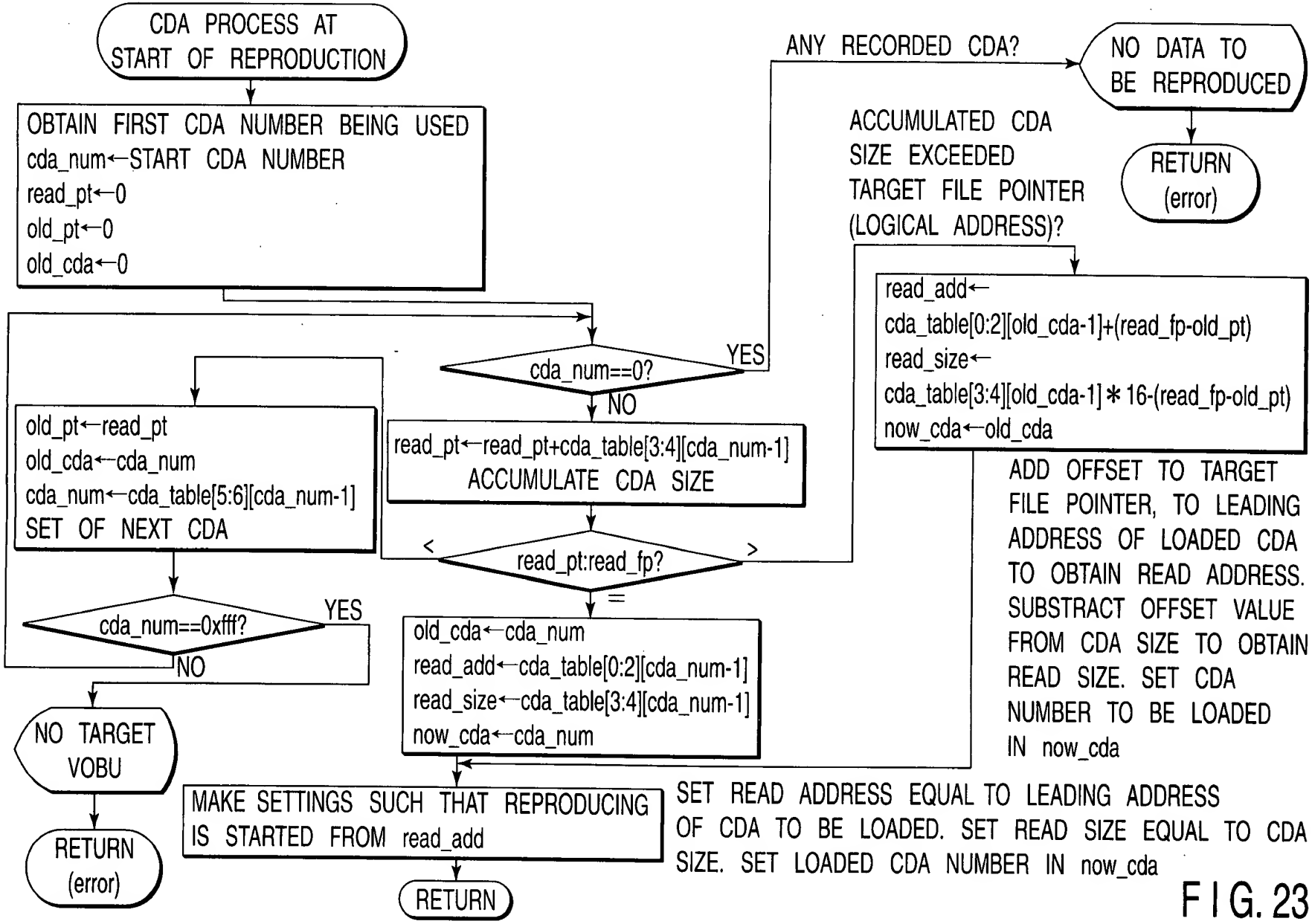

CDA PROCESS AT START OF REPRODUCTION
OBTAIN FIRST CDA NUMBER BEING USED
cda_num←START CDA NUMBER
read_pt←0
old_pt←0
old_cda←0
cda_num==0?
YES
NO
ANY RECORDED CDA?
NO DATA TO BE REPRODUCED
RETURN (error)
ACCUMULATED CDA SIZE EXCEEDED TARGET FILE POINTER (LOGICAL ADDRESS)?
read_add←
cda_table[0:2][old_cda-1]+(read_fp-old_pt)
read_size←
cda_table[3:4][old_cda-1]*16-(read_fp-old_pt)
now_cda←old_cda
ADD OFFSET TO TARGET FILE POINTER, TO LEADING ADDRESS OF LOADED CDA TO OBTAIN READ ADDRESS. SUBSTRACT OFFSET VALUE FROM CDA SIZE TO OBTAIN READ SIZE. SET CDA NUMBER TO BE LOADED IN now_cda
read_pt←read_pt+cda_table[3:4][cda_num-1]
ACCUMULATE CDA SIZE
< read_pt:read_fp? >
=
old_pt←read_pt
old_cda←cda_num
cda_num←cda_table[5:6][cda_num-1]
SET OF NEXT CDA
cda_num==0xfff?
YES
NO
old_cda←cda_num
read_add←cda_table[0:2][cda_num-1]
read_size←cda_table[3:4][cda_num-1]
now_cda←cda_num
NO TARGET VOBU
RETURN (error)
MAKE SETTINGS SUCH THAT REPRODUCING IS STARTED FROM read_add
SET READ ADDRESS EQUAL TO LEADING ADDRESS OF CDA TO BE LOADED. SET READ SIZE EQUAL TO CDA SIZE. SET LOADED CDA NUMBER IN now_cda
RETURN

FIG. 23

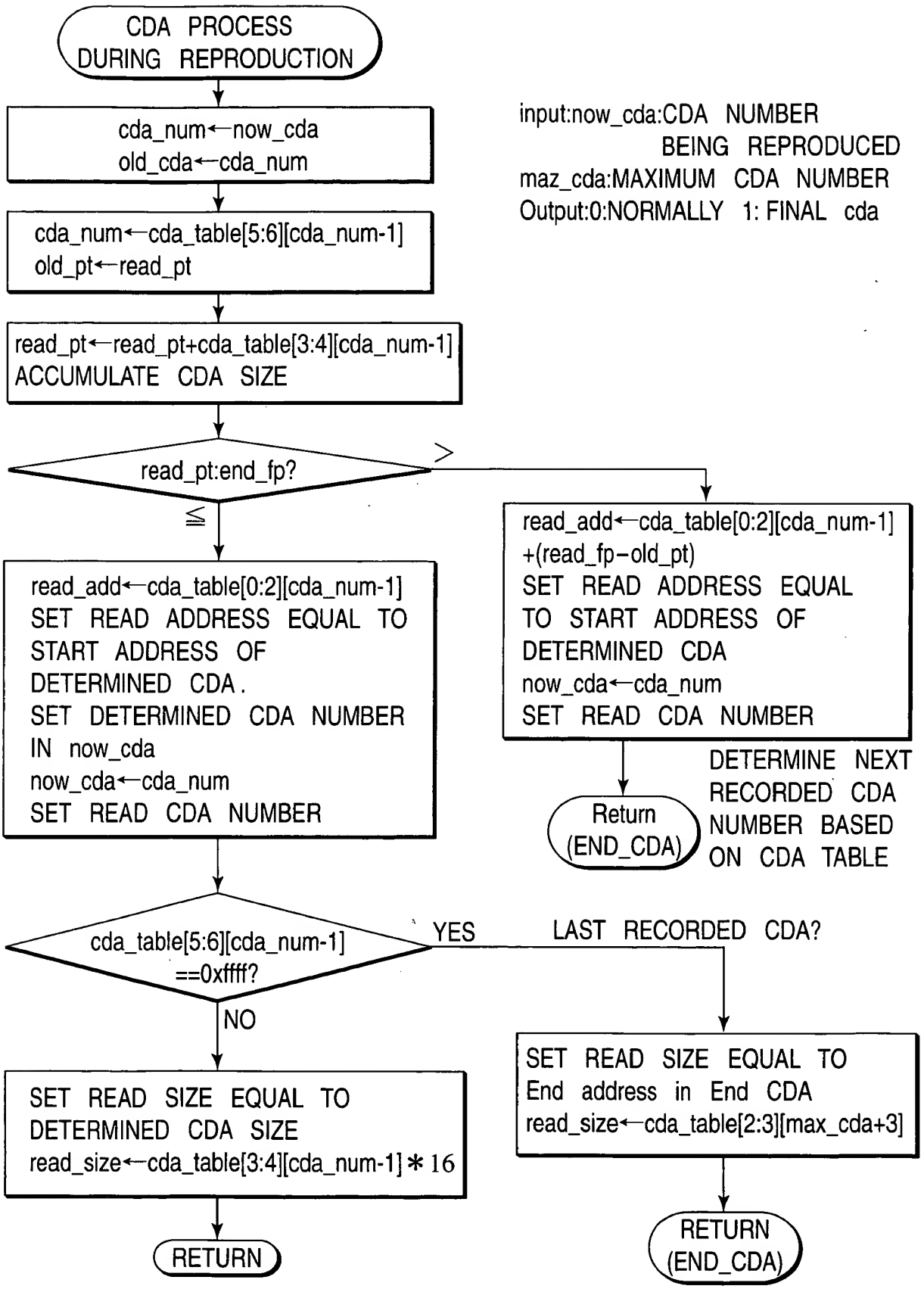

CDA PROCESS DURING REPRODUCTION
input:now_cda:CDA NUMBER BEING REPRODUCED
maz_cda:MAXIMUM CDA NUMBER
Output:0:NORMALLY 1: FINAL cda
cda_num←now_cda
old_cda←cda_num
cda_num←cda_table[5:6][cda_num-1]
old_pt←read_pt
read_pt←read_pt+cda_table[3:4][cda_num-1]
ACCUMULATE CDA SIZE
read_pt:end_fp?
>
≦
read_add←cda_table[0:2][cda_num-1]
+(read_fp−old_pt)
SET READ ADDRESS EQUAL TO START ADDRESS OF DETERMINED CDA
now_cda←cda_num
SET READ CDA NUMBER
DETERMINE NEXT RECORDED CDA NUMBER BASED ON CDA TABLE
Return (END_CDA)
read_add←cda_table[0:2][cda_num-1]
SET READ ADDRESS EQUAL TO START ADDRESS OF DETERMINED CDA.
SET DETERMINED CDA NUMBER IN now_cda
now_cda←cda_num
SET READ CDA NUMBER
cda_table[5:6][cda_num-1] ==0xffff?
YES
LAST RECORDED CDA?
NO
SET READ SIZE EQUAL TO DETERMINED CDA SIZE
read_size←cda_table[3:4][cda_num-1]＊16
SET READ SIZE EQUAL TO End address in End CDA
read_size←cda_table[2:3][max_cda+3]
RETURN
RETURN (END_CDA)

FIG. 24

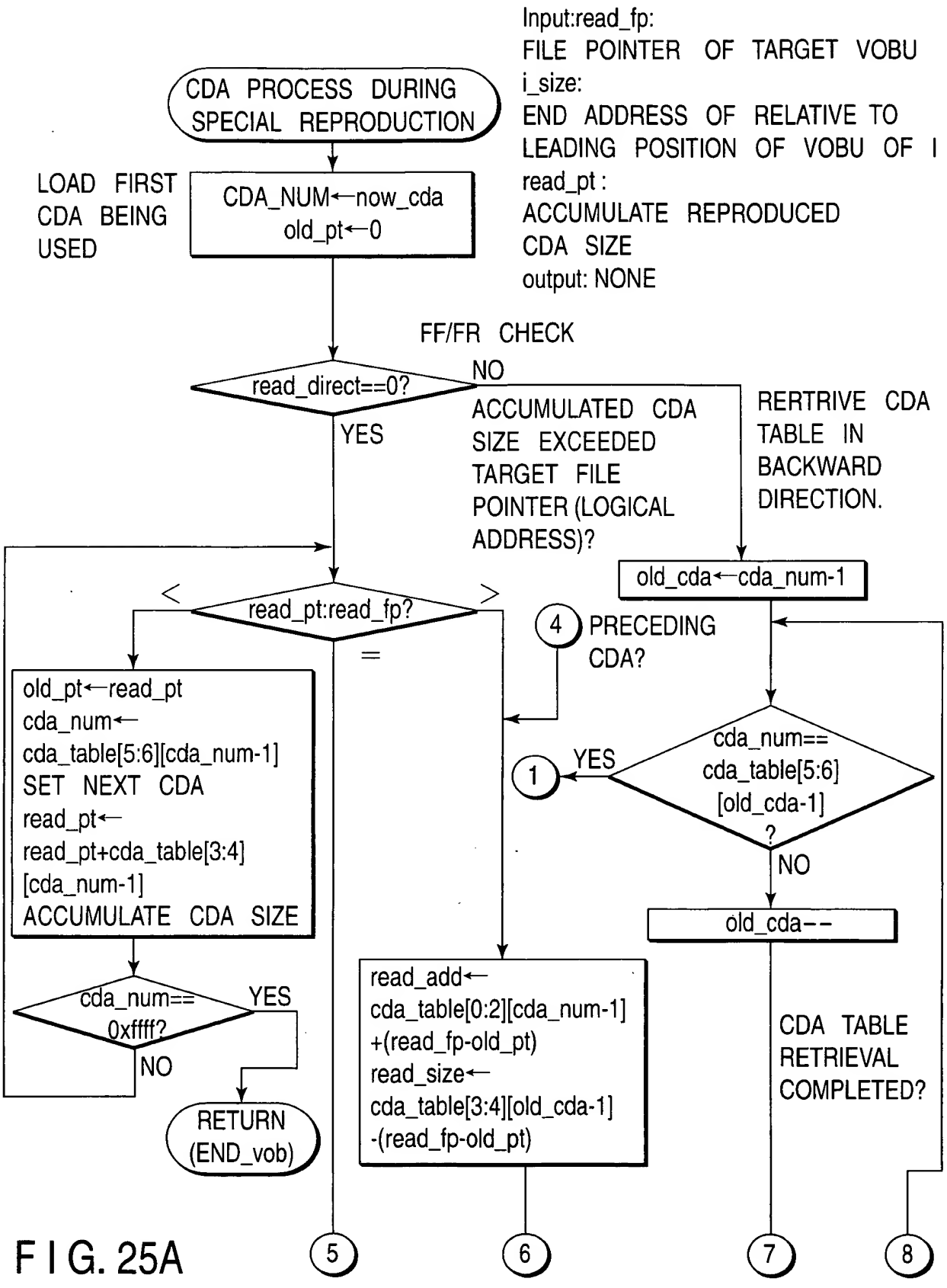

CDA PROCESS DURING SPECIAL REPRODUCTION
Input:read_fp:
FILE POINTER OF TARGET VOBU
i_size:
END ADDRESS OF RELATIVE TO LEADING POSITION OF VOBU OF I
read_pt :
ACCUMULATE REPRODUCED CDA SIZE
output: NONE
LOAD FIRST CDA BEING USED
CDA_NUM←now_cda
old_pt←0
FF/FR CHECK
read_direct==0?
NO
YES
ACCUMULATED CDA SIZE EXCEEDED TARGET FILE POINTER (LOGICAL ADDRESS)?
RERTRIVE CDA TABLE IN BACKWARD DIRECTION.
old_cda←cda_num-1
read_pt:read_fp?
<
=
>
4
PRECEDING CDA?
old_pt←read_pt
cda_num←
cda_table[5:6][cda_num-1]
SET NEXT CDA
read_pt←
read_pt+cda_table[3:4]
[cda_num-1]
ACCUMULATE CDA SIZE
cda_num==
cda_table[5:6]
[old_cda-1]
?
YES
1
NO
old_cda--
cda_num==
0xffff?
YES
NO
read_add←
cda_table[0:2][cda_num-1]
+(read_fp-old_pt)
read_size←
cda_table[3:4][old_cda-1]
-(read_fp-old_pt)
CDA TABLE RETRIEVAL COMPLETED?
RETURN
(END_vob)
5
6
7
8

FIG. 25A

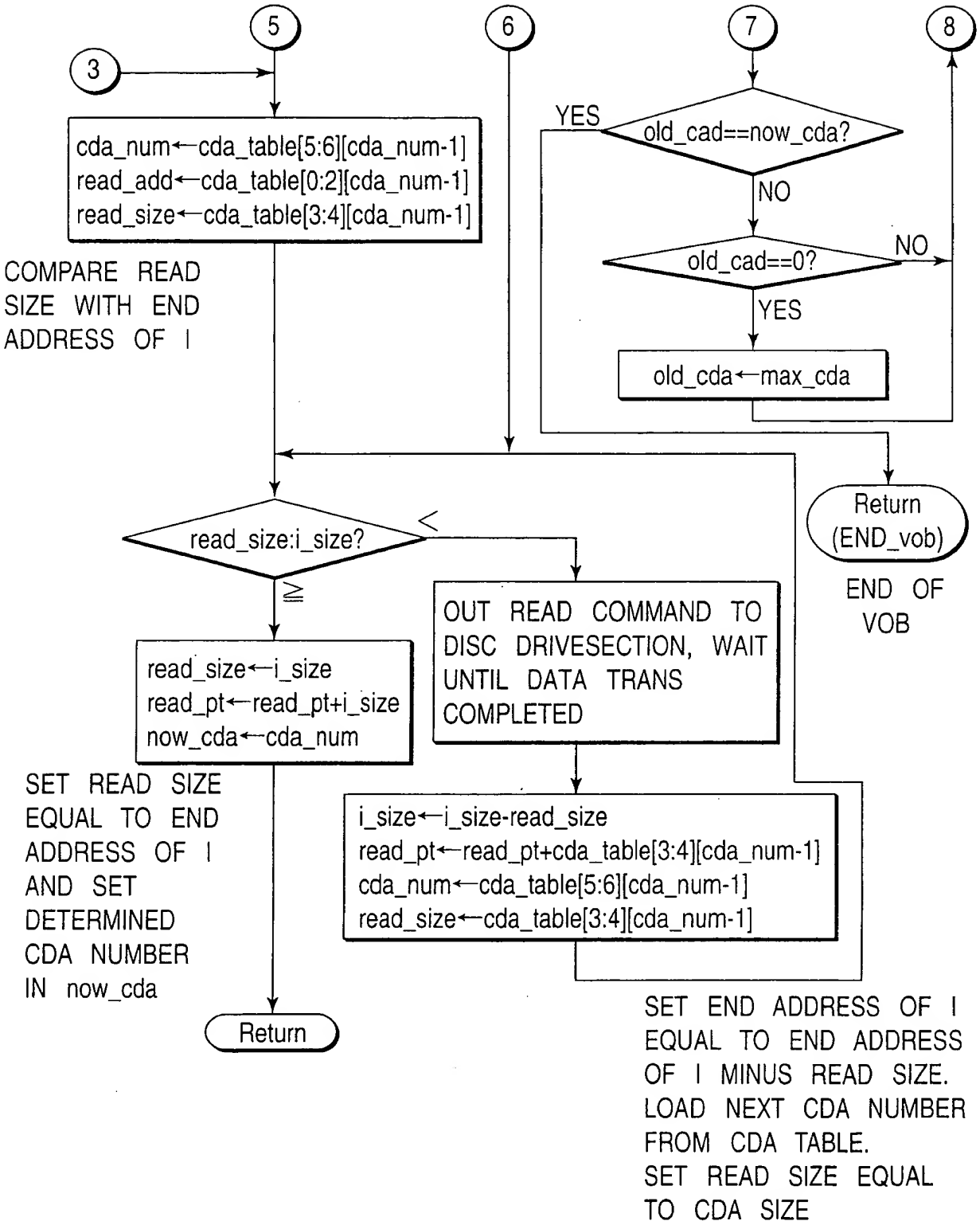

5
6
7
8
3
cda_num←cda_table[5:6][cda_num-1]
read_add←cda_table[0:2][cda_num-1]
read_size←cda_table[3:4][cda_num-1]
COMPARE READ SIZE WITH END ADDRESS OF I
YES
old_cad==now_cda?
NO
old_cad==0?
NO
YES
old_cda←max_cda
Return (END_vob)
END OF VOB
read_size:i_size?
<
≧
OUT READ COMMAND TO DISC DRIVESECTION, WAIT UNTIL DATA TRANS COMPLETED
read_size←i_size
read_pt←read_pt+i_size
now_cda←cda_num
SET READ SIZE EQUAL TO END ADDRESS OF I AND SET DETERMINED CDA NUMBER IN now_cda
i_size←i_size-read_size
read_pt←read_pt+cda_table[3:4][cda_num-1]
cda_num←cda_table[5:6][cda_num-1]
read_size←cda_table[3:4][cda_num-1]
SET END ADDRESS OF I EQUAL TO END ADDRESS OF I MINUS READ SIZE. LOAD NEXT CDA NUMBER FROM CDA TABLE. SET READ SIZE EQUAL TO CDA SIZE
Return

FIG. 25B

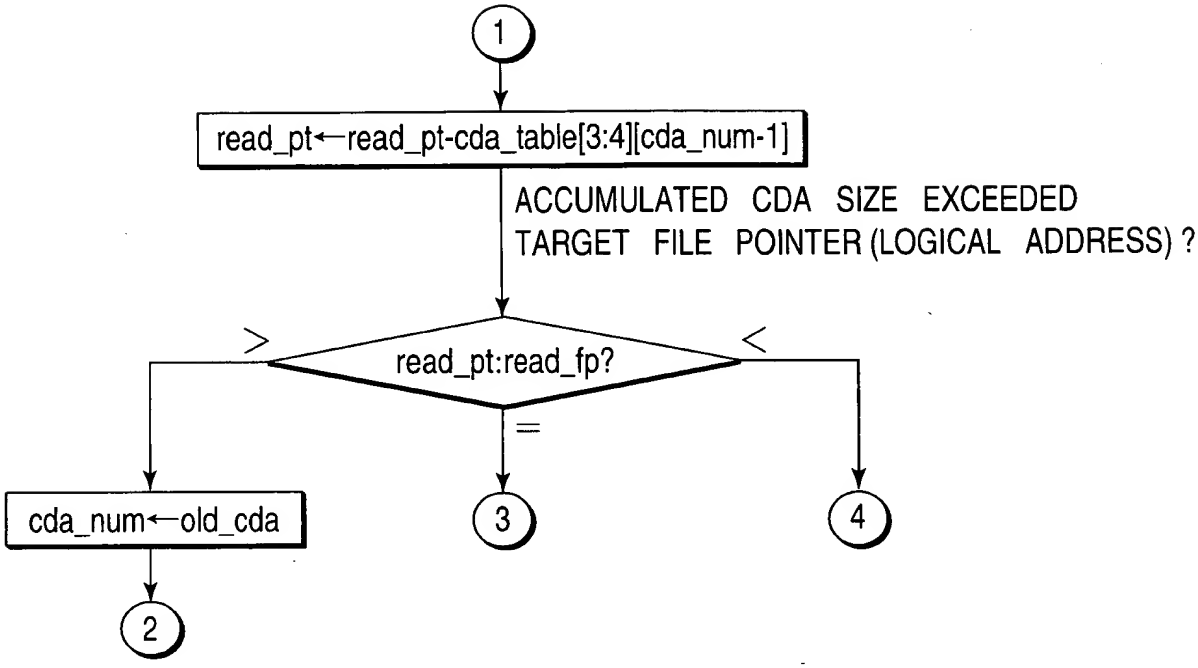

1
read_pt←read_pt-cda_table[3:4][cda_num-1]
ACCUMULATED CDA SIZE EXCEEDED
TARGET FILE POINTER (LOGICAL ADDRESS) ?
read_pt:read_fp?
>
=
<
cda_num←old_cda
2
3
4

FIG. 25C

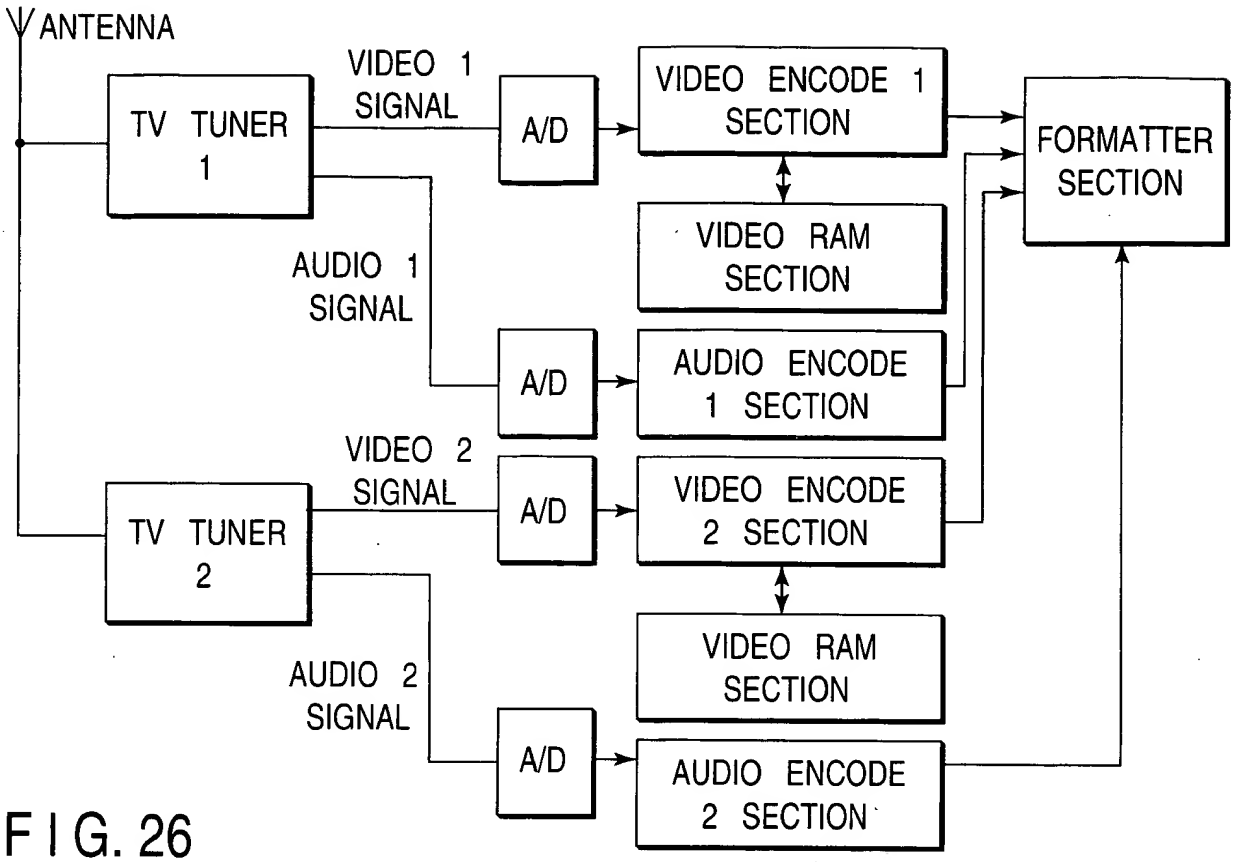

ANTENNA
TV TUNER 1
VIDEO 1 SIGNAL
A/D
VIDEO ENCODE 1 SECTION
FORMATTER SECTION
VIDEO RAM SECTION
AUDIO 1 SIGNAL
A/D
AUDIO ENCODE 1 SECTION
VIDEO 2 SIGNAL
TV TUNER 2
A/D
VIDEO ENCODE 2 SECTION
VIDEO RAM SECTION
AUDIO 2 SIGNAL
A/D
AUDIO ENCODE 2 SECTION

FIG. 26

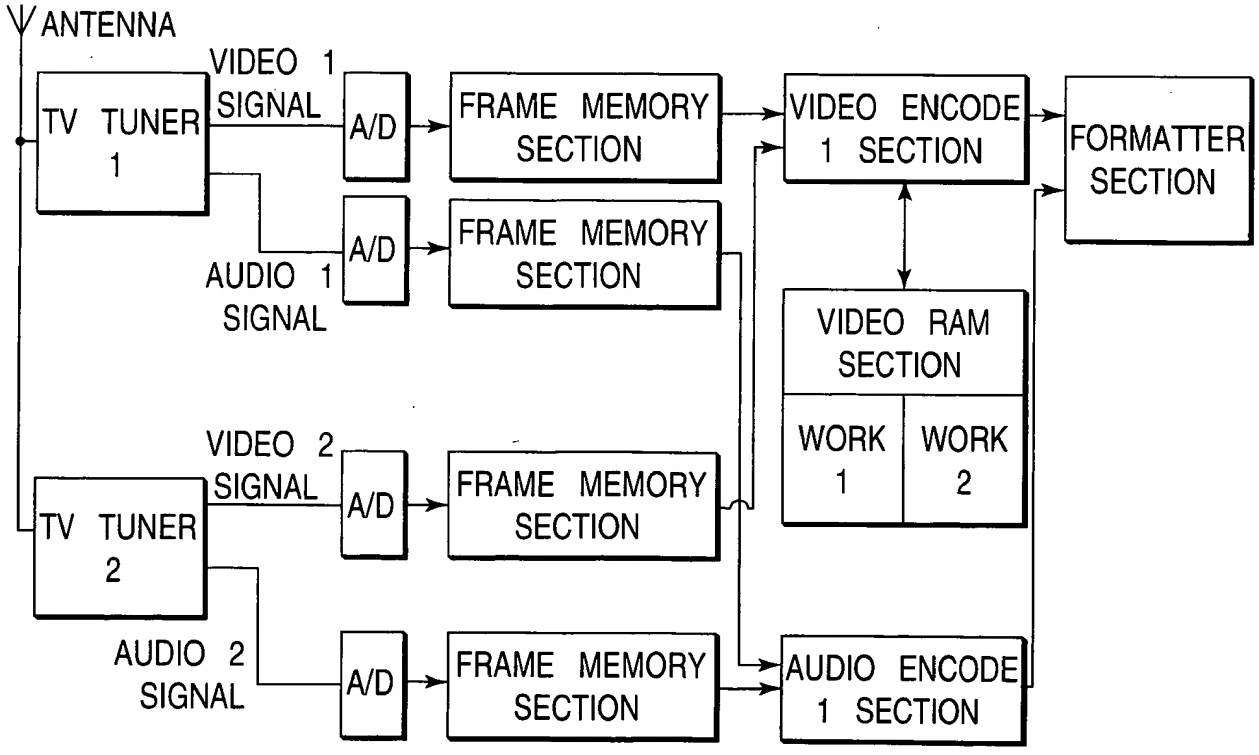

ANTENNA
TV TUNER 1
VIDEO 1 SIGNAL
A/D
FRAME MEMORY SECTION
VIDEO ENCODE 1 SECTION
FORMATTER SECTION
A/D
FRAME MEMORY SECTION
AUDIO 1 SIGNAL
VIDEO RAM SECTION
WORK 1
WORK 2
VIDEO 2 SIGNAL
TV TUNER 2
A/D
FRAME MEMORY SECTION
AUDIO 2 SIGNAL
A/D
FRAME MEMORY SECTION
AUDIO ENCODE 1 SECTION

FIG. 27